ĐẠI HỌC QUỐC GIA

ĐẠI HỌC BÁCH KHOA TP HỒ CHÍ MINH

BÀI TẬP LỚN CÔNG NGHỆ PHẦN MỀM

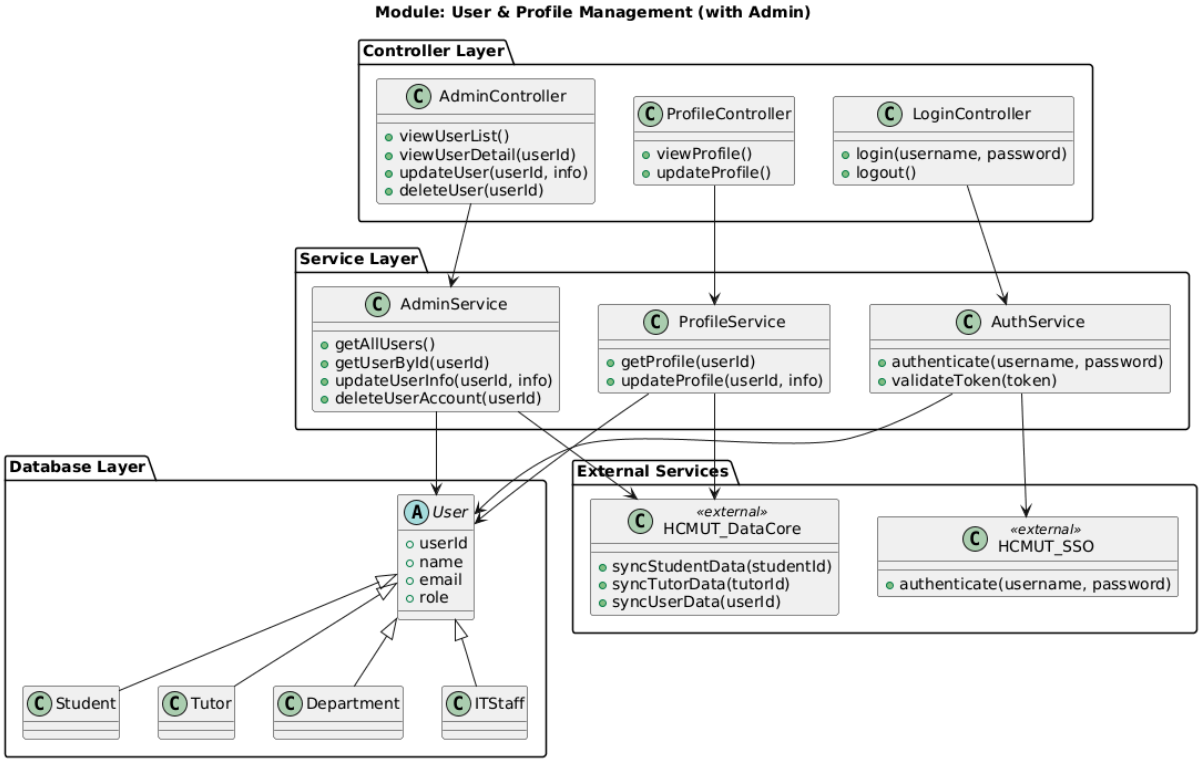
SUBMISSION #3

LỚP: LO3, LO4 - GVHD: thầy Mai Đức Trung - NHÓM: New Convolution L04 :)

STT	MSSV	HỌ	TÊN
1	2310596	Nguyễn Ánh	Dương
2	2312397	Lê Bá	Nguyễn
3	2311628	Nguyễn Trần Đức Duy	Khoa
4	2311770	Mau Gia	Kiệt
5	2310270	Trương Lục Gia	Bảo
6	2310016	Nguyễn Khúc Thái	An
7	2310013	Nghiêm Nguyễn Trường	An

I. Class Diagram:

1. Module Authentication - Profile Management

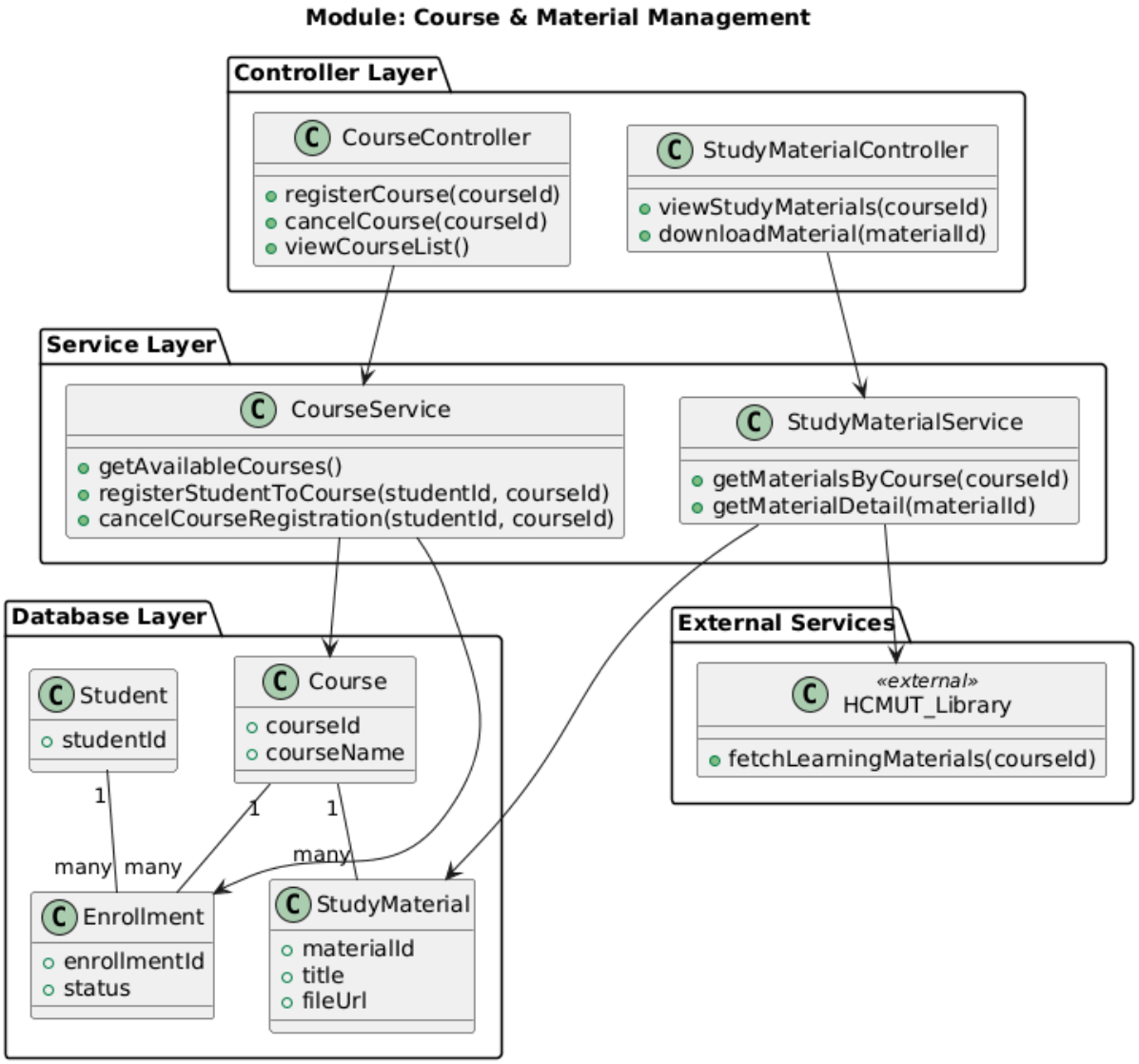


Giải thích về các phương thức:

- **LoginController**
  - **login(username, password)**: Nhận tên đăng nhập và mật khẩu từ giao diện người dùng, sau đó gọi phương thức authenticate của AuthService để xử lý việc xác thực. Nó sẽ nhận kết quả và hiển thị trang tiếp theo hoặc thông báo lỗi.
  - **logout()**: Xử lý yêu cầu đăng xuất, xóa thông tin phiên làm việc (session) của người dùng và chuyển hướng về trang đăng nhập.
- **ProfileController**
  - **viewProfile()**: Lấy thông tin người dùng đang đăng nhập, gọi ProfileService để lấy chi tiết hồ sơ và hiển thị lên giao diện cho người dùng xem.
  - **updateProfile()**: Nhận thông tin mới mà người dùng muốn cập nhật từ giao diện, sau đó gửi thông tin này đến ProfileService để lưu lại.
- **AuthService**
  - **authenticate(username, password)**: Gọi đến dịch vụ ngoài HCMUT_SSO để kiểm tra thông tin đăng nhập có hợp lệ không. Nếu hợp lệ, nó tạo ra phiên làm việc cho người dùng trong hệ thống.
  - **validateToken(token)**: Kiểm tra xem một token xác thực có còn hợp lệ hay không mỗi khi người dùng truy cập một tài nguyên cần bảo vệ.
- **ProfileService**
  - **getProfile(userId)**: Lấy thông tin chi tiết của người dùng từ cơ sở dữ liệu (class User).

    + **updateProfile(userId, info)**: Nhận thông tin mới, kiểm tra tính hợp lệ của dữ liệu, và sau đó cập nhật vào cơ sở dữ liệu của hệ thống (bảng User).
- **AdminControler:**
    + **viewUserList()**: Gọi service để lấy danh sách tất cả người dùng và hiển thị cho Admin.
    + **viewUserDetail(userId)**: Lấy và hiển thị thông tin chi tiết của một người dùng cụ thể.
    + **updateUser(userId, info):** Gửi yêu cầu cập nhật thông tin cho một người dùng bất kỳ đến service
    + **deleteUser(userId)**: Gửi yêu cầu xóa một tài khoản người dùng đến service.
- AdminService:
    + **getAllUsers():** Truy vấn CSDL để lấy tất cả bản ghi trong bảng User
    + **updateUserInfo(userId, info):** Thực hiện logic cập nhật thông tin người dùng trong CSDL
    + **deleteUserAccount(userId)**: Thực hiện logic xóa (hoặc vô hiệu hóa) tài khoản người dùng

2. Module: Course - Material Management



Giải thích về các phương thức:

- **CourseController**
  - + **registerCourse(courseId):** Nhận yêu cầu đăng ký một khóa học từ sinh viên. Lấy studentId của sinh viên đang đăng nhập và courseId từ yêu cầu, sau đó gọi CourseService để thực hiện việc đăng ký.
  - + **cancelCourse(courseId)**: Tương tự như **regisster Course**, nhưng để xử lý yêu cầu hủy đăng ký một khóa học.
  - + **viewCourseList()**: Gọi CourseService để lấy danh sách tất cả các khóa học đang có và hiển thị ra cho người dùng.

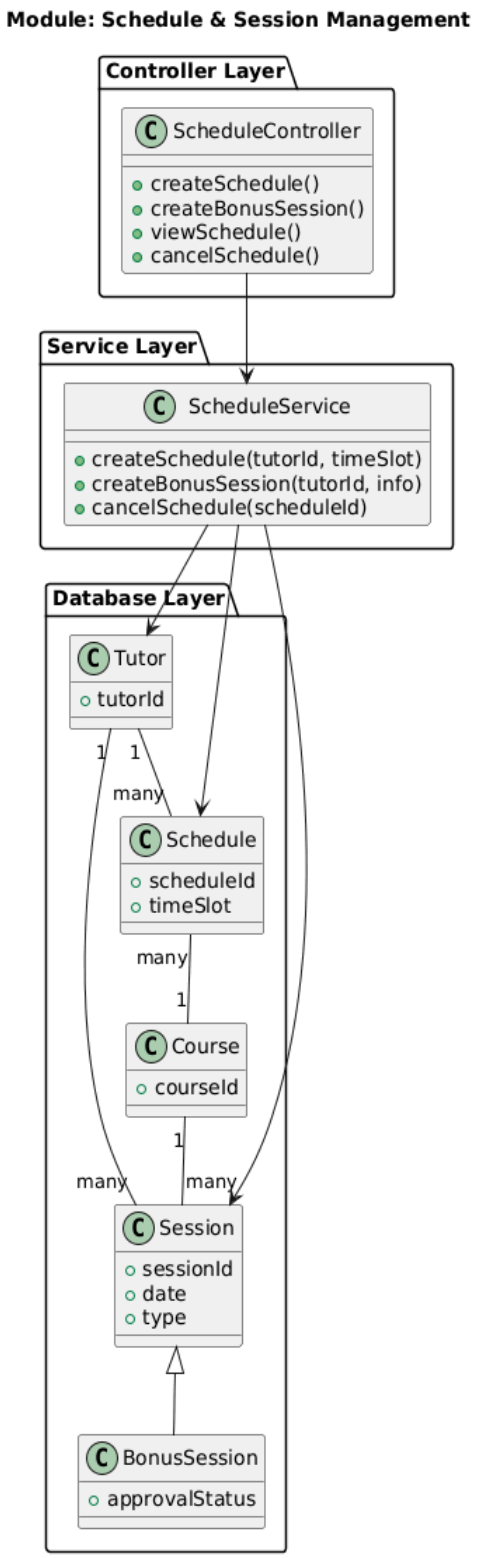
- **StudyMaterialController**
  - + **viewStudyMaterials(courseId)**: Gọi StudyMaterialService để lấy danh sách tài liệu của một khóa học cụ thể và hiển thị chúng.

    + **downloadMaterial(materialId)**: Xử lý yêu cầu tải về một tài liệu cụ thể. Gọi service để lấy đường dẫn file và trả về cho trình duyệt để tải xuống.

- **CourseService**
    + **getAvailableCourses()**: Truy vấn cơ sở dữ liệu để lấy tất cả các bản ghi trong bảng Course.
    + **registerStudentToCourse(studentId, courseId)**: Thực hiện logic nghiệp vụ đăng ký. Kiểm tra các điều kiện, sau đó tạo một bản ghi mới trong bảng Enrollment để liên kết Student và Course.
    + **cancelCourseRegistration(studentId, courseId)**: Tìm bản ghi Enrollment tương ứng và cập nhật trạng thái của nó thành "đã hủy" hoặc xóa bản ghi đó.

- **StudyMaterialService**
    + **getMaterialsByCourse(courseId)**: Truy cập bảng StudyMaterial để lấy các tài liệu thuộc về một courseId. Nó cũng có thể gọi đến dịch vụ ngoài HCMUT_Library để lấy thêm các tài liệu từ thư viện trường và tổng hợp lại.
    + **getMaterialDetail(materialId)**: Lấy thông tin chi tiết (như URL) của một tài liệu duy nhất từ CSDL.

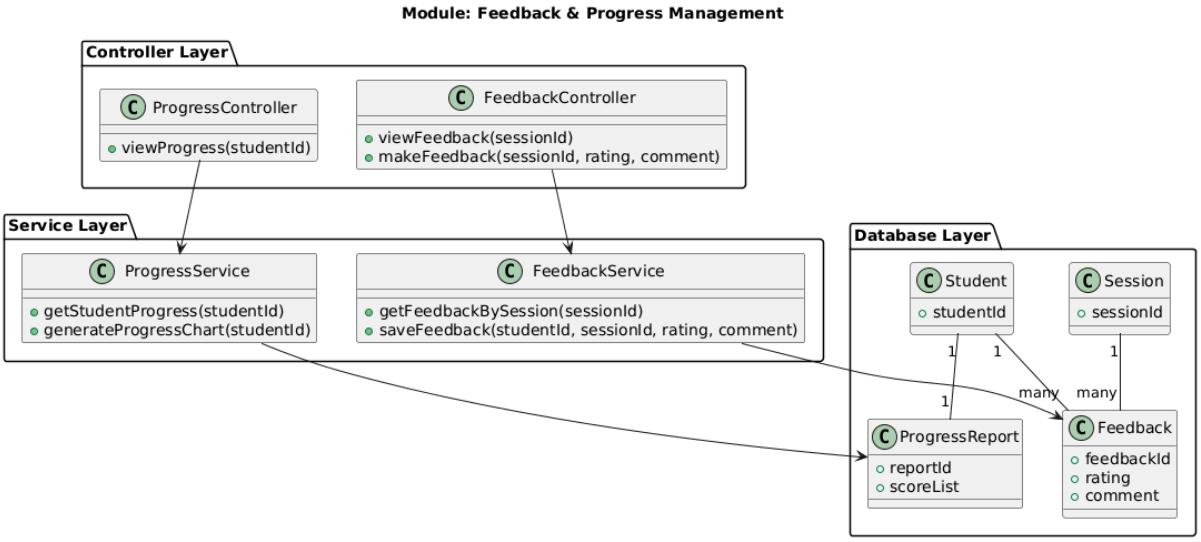
3. Module: Schedule - Session Management



Giải thích về các phương thức:

- **ScheduleController**
  - + **createSchedule()**: Nhận yêu cầu tạo lịch mới từ gia sư và gọi ScheduleService.
  - + **createBonusSession()**: Xử lý yêu cầu tạo một buổi học bổ sung (có thể có các quy tắc khác, ví dụ cần phê duyệt).
  - + **viewSchedule()**: Lấy lịch trình của gia sư từ ScheduleService và hiển thị.
  - + **cancelSchedule()**: Xử lý yêu cầu hủy một lịch đã đặt.
- **ScheduleService**
  - + **createSchedule(tutorId, timeSlot)**: Tạo một bản ghi mới trong bảng Schedule và có thể cả các bản ghi Session tương ứng. Kiểm tra logic để đảm bảo không bị trùng lịch.
  - + **createBonusSession(tutorId, info)**: Tạo một bản ghi BonusSession đặc biệt, có thể có trạng thái ban đầu là "chờ phê duyệt".
  - + **cancelSchedule(scheduleId)**: Tìm lịch trình trong CSDL và cập nhật trạng thái của nó thành "đã hủy".

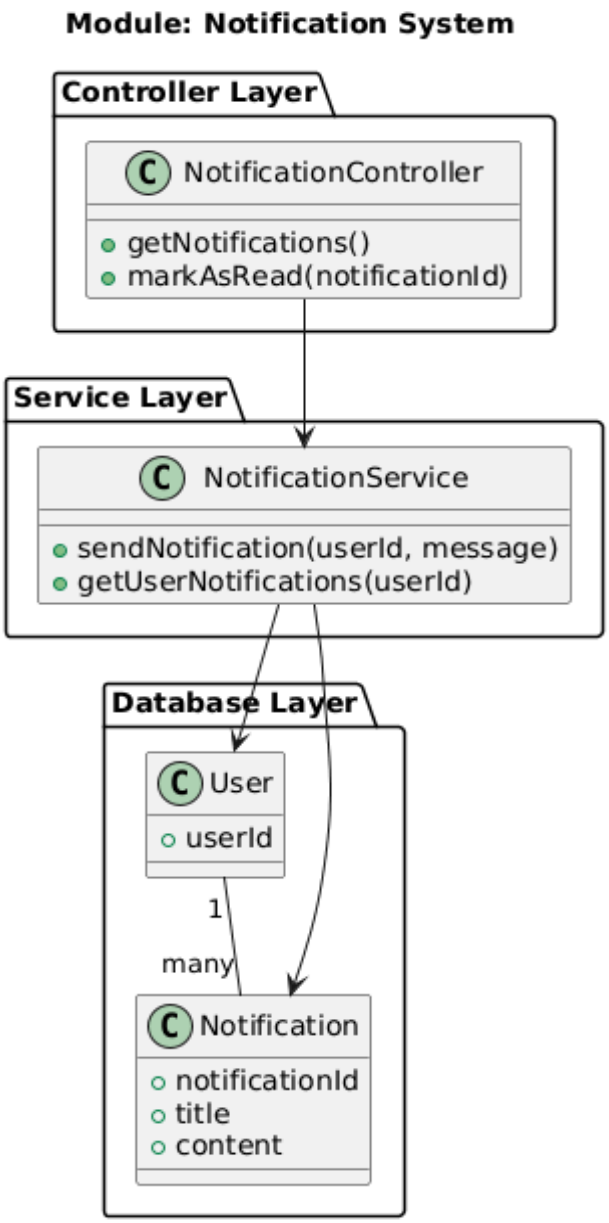
4. Module: Feedback - Progress Management



Giải thích về các phương thức:

- **FeedbackController**
  - **viewFeedback(sessionId)**: Lấy tất cả phản hồi của một buổi học và hiển thị (thường dành cho gia sư hoặc quản lý).
  - **makeFeedback(sessionId, rating, comment)**: Nhận nội dung phản hồi (đánh giá, bình luận) từ sinh viên và gửi đến FeedbackService để lưu lại.
- **ProgressController**
  - **viewProgress(studentId)**: Gọi ProgressService để lấy dữ liệu tiến độ học tập của sinh viên và hiển thị.
- **FeedbackService**
  - **getFeedbackBySession(sessionId)**: Truy vấn CSDL để lấy tất cả các bản ghi Feedback liên quan đến một sessionId.
  - **saveFeedback(studentId, sessionId, rating, comment)**: Tạo một bản ghi Feedback mới trong CSDL, liên kết nó với sinh viên và buổi học tương ứng.
- **ProgressService**
  - **getStudentProgress(studentId):** Lấy dữ liệu từ bảng ProgressReport. Logic này có thể phức tạp, bao gồm việc tổng hợp điểm số, tính toán tỷ lệ hoàn thành...
  - **generateProgressChart(studentId)**: Lấy dữ liệu tiến độ và xử lý nó thành một định dạng mà thư viện vẽ biểu đồ trên giao diện có thể hiểu và vẽ ra được.

5. Module: Notification System

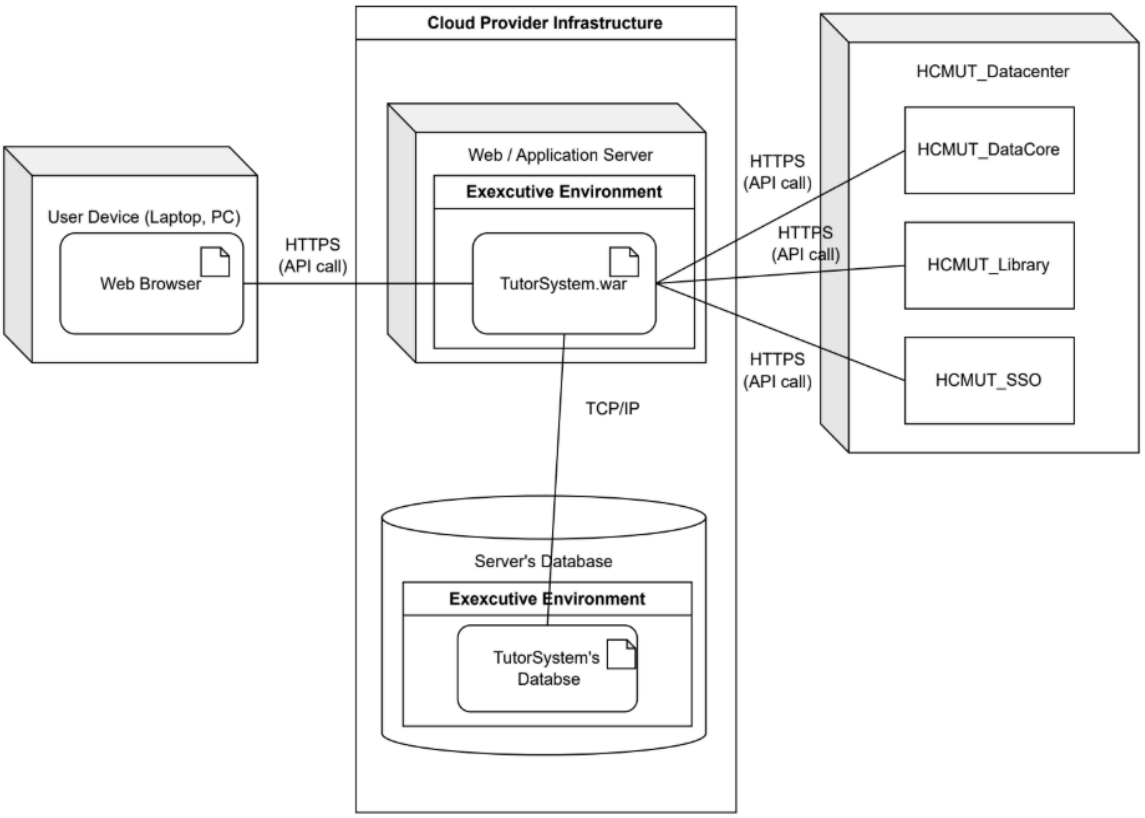


Giải thích về các phương thức:

- **NotificationController**
  - **getNotifications()**: Gọi NotificationService để lấy danh sách thông báo (chưa đọc) cho người dùng hiện tại.
  - **markAsRead(notificationId)**: Xử lý khi người dùng click vào một thông báo, gọi service để cập nhật trạng thái "đã đọc" cho nó.

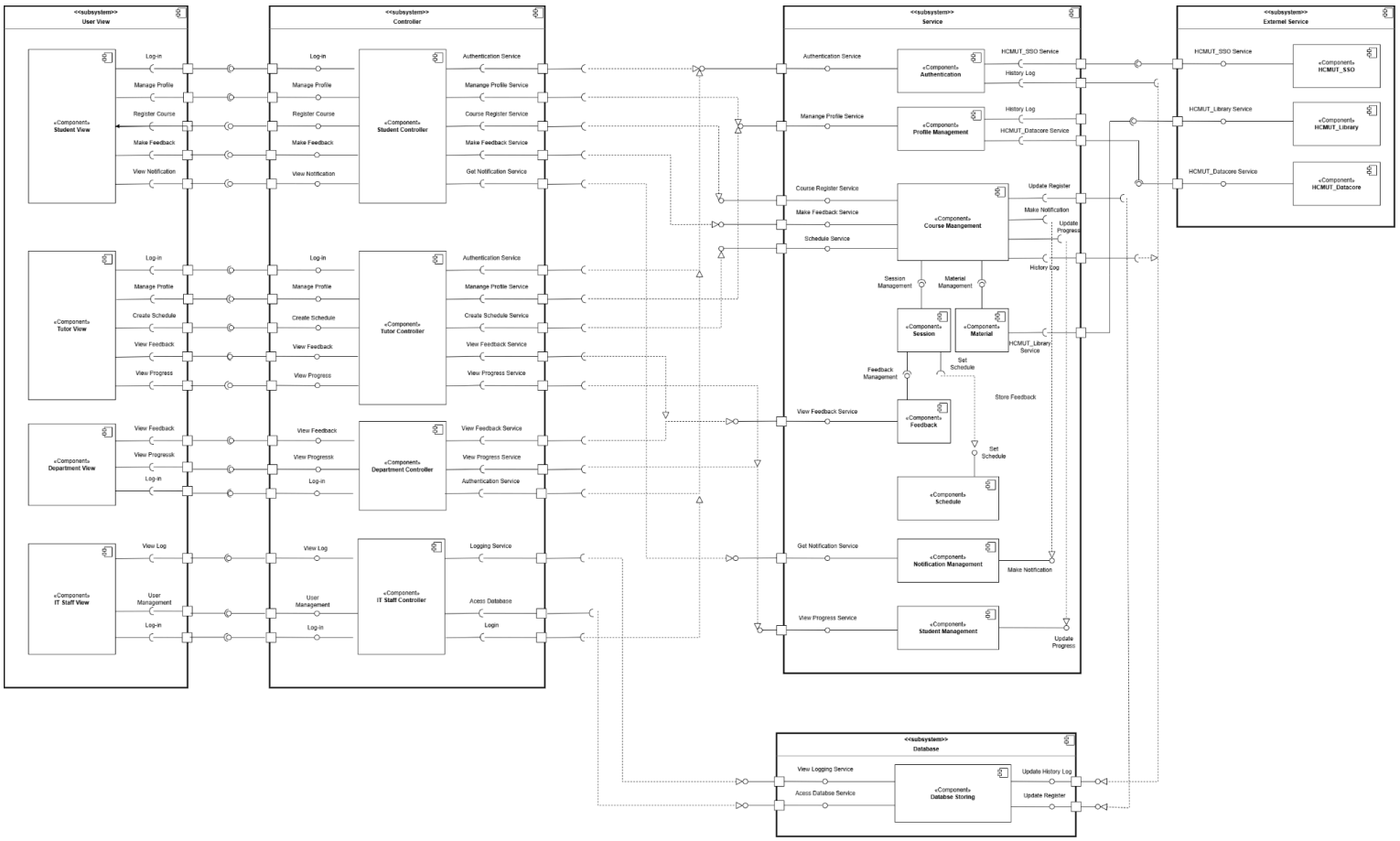
- **NotificationService**
  - **sendNotification(userId, message)**: Đây là một phương thức quan trọng được các service khác gọi đến. Ví dụ, khi CourseService hủy một khóa học, nó sẽ gọi sendNotification để tạo một thông báo trong CSDL và gửi đến tất cả sinh viên.
  - **getUserNotifications(userId)**: Truy cập CSDL để lấy tất cả các bản ghi Notification thuộc về một userId.

II. Deployment View:



- Mô tả về Deployment View: mở Web Browser trên User Device và truy cập vào địa chỉ web của hệ thống. Trình duyệt gửi một yêu cầu qua mạng Internet (sử dụng giao thức HTTPS) đến Web/Application Server đang chạy trên Cloud. TutorSystem.war nhận yêu cầu. Ứng dụng backend này sẽ gửi một truy vấn qua mạng nội bộ của Cloud (sử dụng giao thức TCP/IP) đến TutorSystem Database. Tutor System Database trên server đó xử lý truy vấn và trả kết quả về. Ứng dụng backend nhận dữ liệu, định dạng lại và gửi trả về cho Web Browser của sinh viên. Backend sẽ gọi một truy vấn HTTPS tới HCMUT_Datacore nếu cần truy vấn các thông tin (xác thực đăng nhập, lấy tài liệu,...)

III. Component Diagram:



1. Subsystem User View và Subsystem Controller:

- Phần Subsystem User View mô tả UI của người dùng.
- Phần Subsystem Controller đóng vai trò điều hướng yêu cầu từ UI.
- Component Student View và Student Controller: Bao gồm các Required Component khác như Đăng nhập, Quản lý hồ sơ, Đăng ký khóa học, Make Feedback và Xem thông báo. Tất cả các Required Component này được chuyển đến Provided Component khác ở Controller để đáp ứng các yêu cầu từ người dùng. Student Controller sẽ gọi đến các Provided Component của Subsystem Service để đáp ứng các nhiêu cầu này, vì thế Subsystem Controller đóng vai trò trung gian giữa View và Service, để điều hướng yêu cầu từ người dùng. Tương tự cho các Component khác. Component IT Staff Controller thì có phần Required Component phụ thuộc thẳng đến Subsystem Database, cụ thể là Access Database và Logging Service (để xem lịch sử log).

2. **Subsystem Service:**
    - Component Service đóng vai trò xử lý logic các yêu cầu từ người dùng được chuyển tiếp từ Controller.
    - Component Authentication: Đóng vai trò xử lý xác thực, đáp ứng các yêu cầu xác thực khi đăng nhập, và Required Component là HCMUT_SSO Service.
    - Component Profile Management: Đóng vai trò xử lý logic các tác vụ liên quan đến quản lý hồ sơ cá nhân của người dùng, và Required Component là HCMUT_Datacore để cập nhật lên hệ thống trường.
    - Component Course Management: Xử lý các logic cụ thể như Đăng ký khóa học từ Student, Quản lý lên lịch từ Tutor cũng như những tác vụ liên quan đến Feedback.
        + Gọi tới Component Session để quản lý một buổi học, nó cũng chịu trách nhiệm cho việc quản lý Feedback từ sinh viên, cũng như lưu trữ các Feedback đó để được xem bởi Tutor hoặc Department. Session cũng quản lý chính việc quản lý lên lịch của Tutor.
        + Component Material đóng đáp ứng nhu cầu lấy tài liệu, nên Required Component là HCMUT Library.
        + Component Notification Management quản lý thông báo mới, đáp ứng nhu cầu thông báo của Course Management.
        + Component Student Management đóng vai trò quản lý các sinh viên đang tham gia khóa học, đáp ứng nhu cầu xem được tiến độ học tập của Sinh viên cho Tutor hoặc Bộ môn.

3. **Subsystem External Service:**
    - Là subsystem bao gồm các component: HCMUT_SSO, HCMUT_Library và HCMUT_Datacore.
    - Subsystem này chủ yếu là của trường, đáp ứng các nhu cầu khi cần xác thực đăng nhập, quản lý hồ sơ cá nhân hay truy cập lấy tài liệu.

4. **Subsystem Database:**
    - Là nơi lưu trữ các thông tin nội bộ của hệ thống, như thông tin khóa học, thông tin sinh viên đăng ký, lịch sử hoạt động của hệ thống cho việc bảo trì.
    - Gồm 1 Component là Database Storing, có 4 Required Component là View Logging History, Access Database, Update History Log và Update Register.