**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Website học trực tuyến**

**PHẠM HỒNG SƠN**

son.ph173344@sis.hust.edu.vn

**Ngành Công nghệ thông tin**

**Chuyên ngành …**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | ThS. Vũ Đức Vượng |
| **Bộ môn:** | Công nghệ phần mềm |
| **Viện:** | Công nghệ thông tin – Truyền thông |
| **HÀ NỘI, 03/2021** | |

# Lời cam kết

Họ và tên sinh viên: Phạm Hồng Sơn

Điện thoại liên lạc: Phạm Hồng Sơn Email: son.ph173344@sis.hust.edu.vn

Lớp: CNTT-09 Hệ đào tạo: Cử nhân kỹ thuật

Tôi – *Phạm Hồng Sơn* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *ThS. Vũ Đức Vương*. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm 2021*  Tác giả ĐATN  *Phạm Hồng Sơn* |

# Danh mục hình vẽ

[**Hình 1** Biểu đồ use case tổng quan](#bookmark=id.cron8lmhwdpy) 5

[**Hình 2** Biểu đồ use case phân rã Authenticate](#bookmark=id.ki2onvdfwca7) 6

[**Hình 3** Biểu đồ use case phân rã View course list](#bookmark=id.2gt3i2gc14xe) 7

[**Hình 4** Biểu đồ use case phân rã Create section](#bookmark=id.e51s08x99p9b) 8

[**Hình 5** Sơ đồ tổng quan các công nghệ sử dụng 11](#bookmark=id.3u2xrprn59ut)

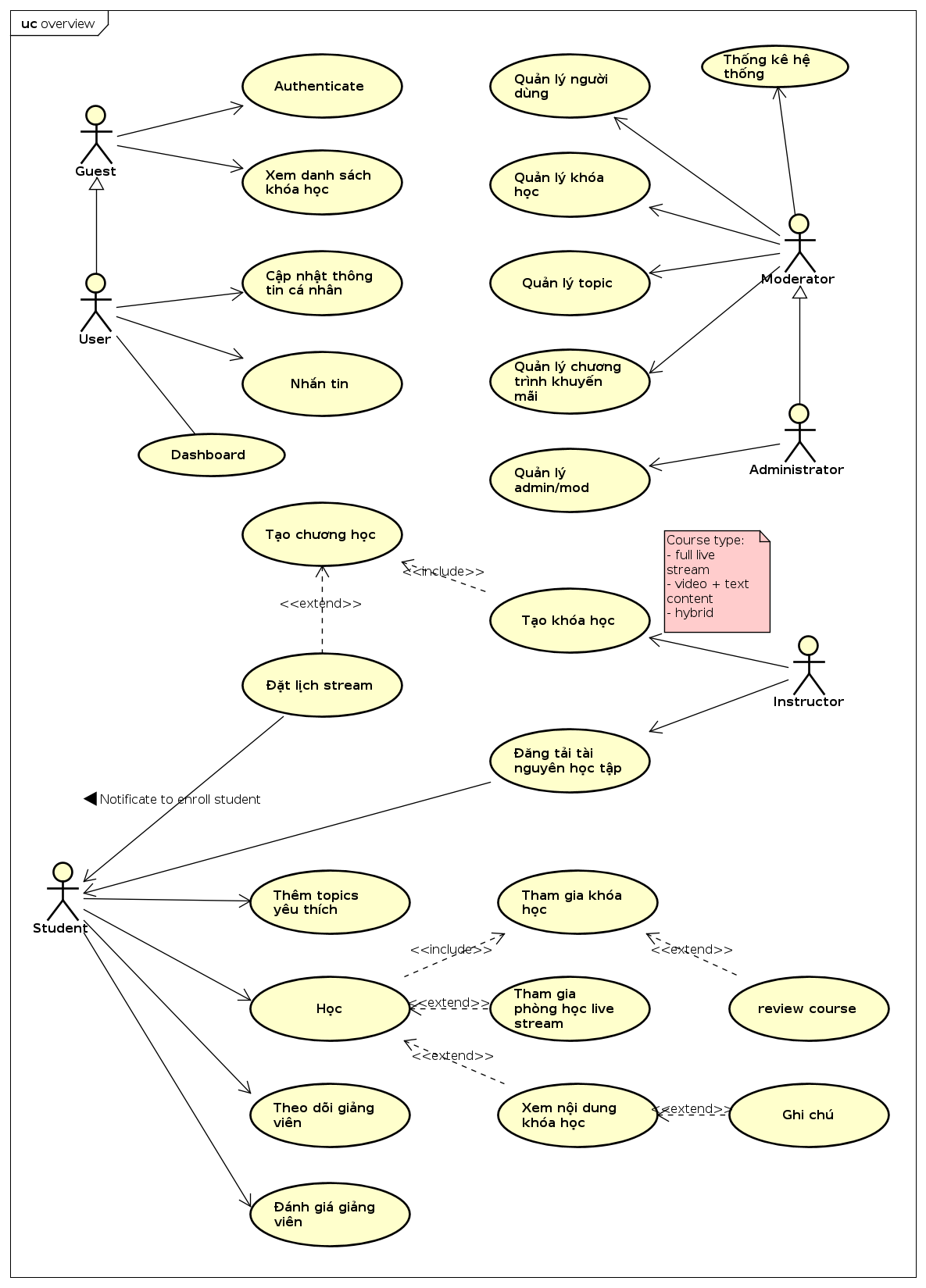
[**Hình 6** Sơ đồ công nghệ sử dụng cho phòng học trực tuyến 11](#bookmark=id.s0hhv0xovr1x)

[**Hình 7** Sơ đồ cơ chế xác thực người dùng 12](#bookmark=id.gomgve4uu61k)

# Chương 1 Khảo sát và phân tích yêu cầu

## Tổng quan chức năng

### Biểu đồ use case tổng quan



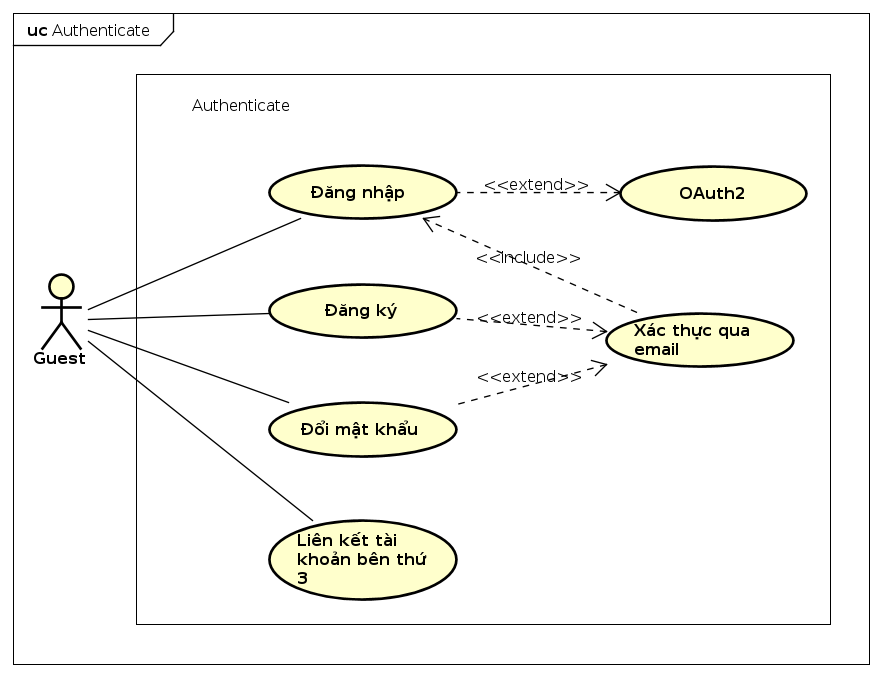
**Hình 1** Biểu đồ use case tổng quan

* Khách (Guest) là những người chưa đăng nhập. Guest có thể tìm kiếm, xem danh sách khóa học, thông tin giới thiệu và đánh giá người dùng của khóa học.
* Người dùng (User) là những người đăng nhập vào hệ thống. User bao gồm Học sinh (Student), Giảng viên (Instructor), Quản trị viên (Administrator/Admin) và Điều hành viên (Moderator/Mod).
* Admin và Mod là những tài khoản đặc biệt của hệ thống, là bên quản lý hệ thống và đưa ra thông báo cho người dùng. Admin có quyền cao hơn Mod và quản lý các tài khoản Mod.

Hệ thống cung cấp nền tảng để Instructor đăng tải các khóa học bằng video, nội dung text hoặc các khóa học hoàn toàn livestream trực tuyến cho các Student. Hệ thống cũng cung cấp một môi trường giao tiếp, trao đổi, giải đáp thắc mắc giữa các User trong các khóa học, bài học và tin nhắn riêng. Hệ thống có cơ chế đánh giá giảng viên, khóa học, các topic giúp Student dễ dàng tìm kiếm, chọn lựa khóa học phù hợp.

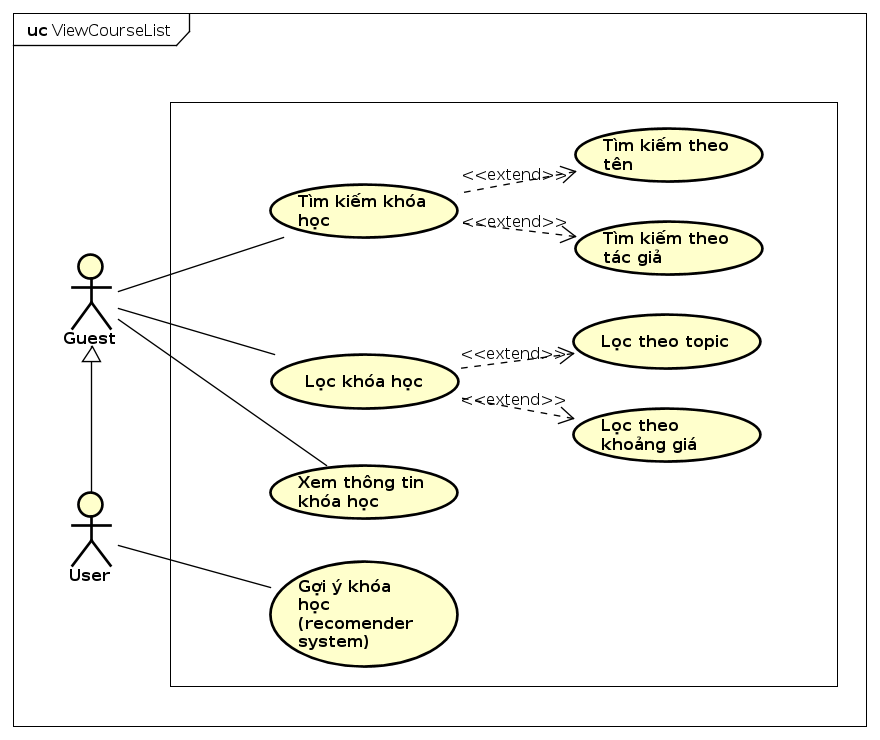
### Biểu đồ use case phân rã

* + - 1. Biểu đồ use case phân rã Authentication



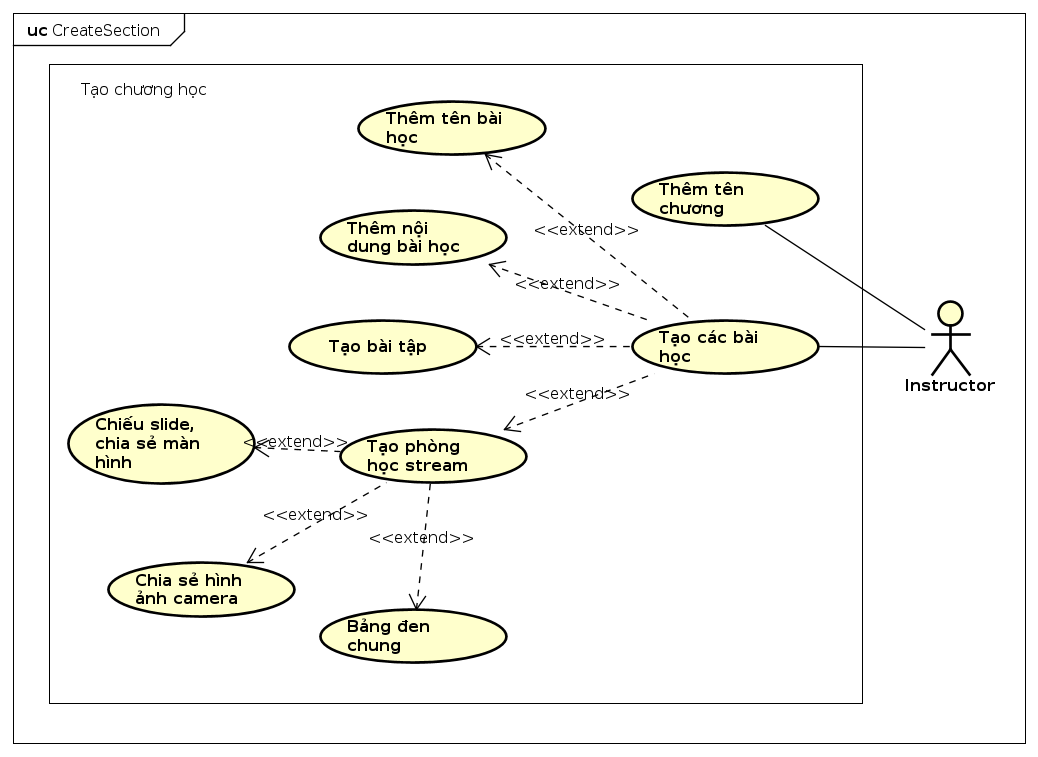
**Hình 2.** Biểu đồ use case phân rã Authenticate

* + - 1. Biểu đồ use case phân rã Xem danh sách khóa học



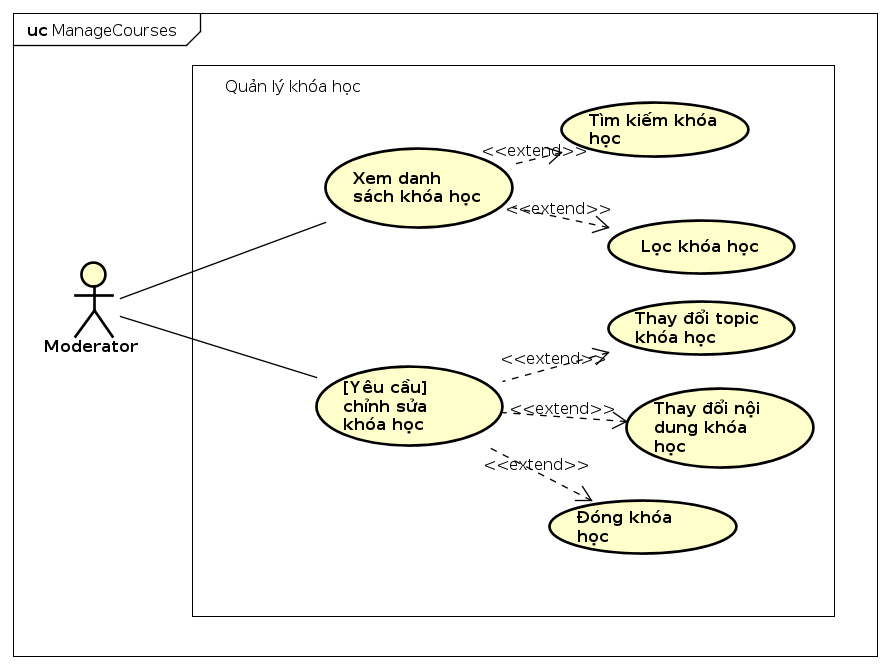
**Hình 3.** Biểu đồ use case phân rã View course list

* + - 1. Biểu đồ use case phân rã tạo chương học



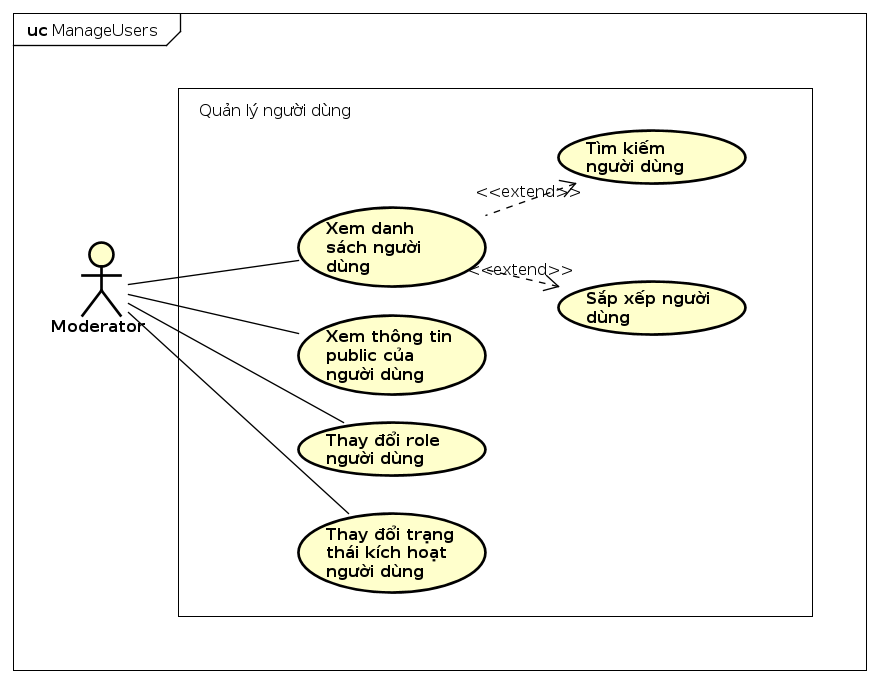
**Hình 4.** Biểu đồ use case phân rã tạo chương học

* + - 1. Biểu đồ use case quản lý khóa học



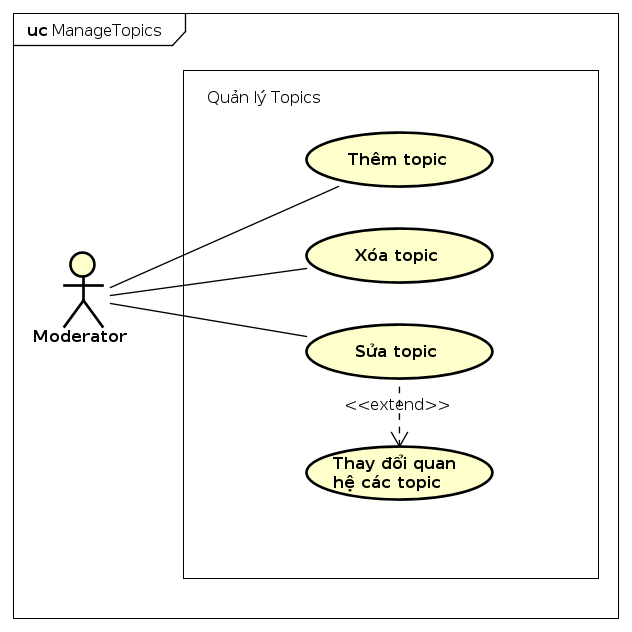
**Hình 5.** Biểu đồ use case phân rã quản lý khóa học

* + - 1. Biểu đồ use case phân rã tạo chương học



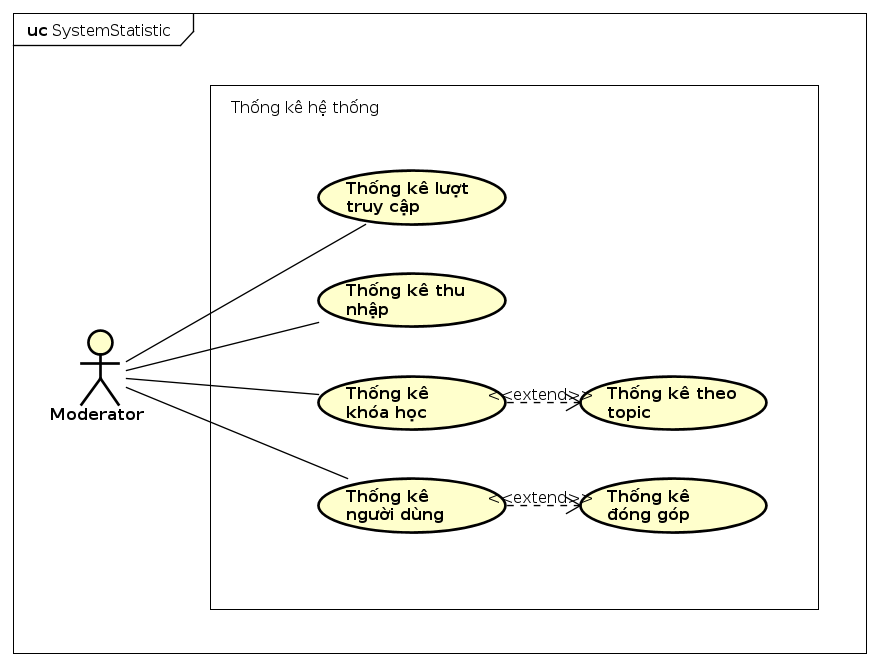
**Hình 6.** Biểu đồ use case phân rã quản lý người users

* + - 1. Biểu đồ use case phân rã quản lý topics



**Hình 7.** Biểu đồ use case phân rã quản lý người users

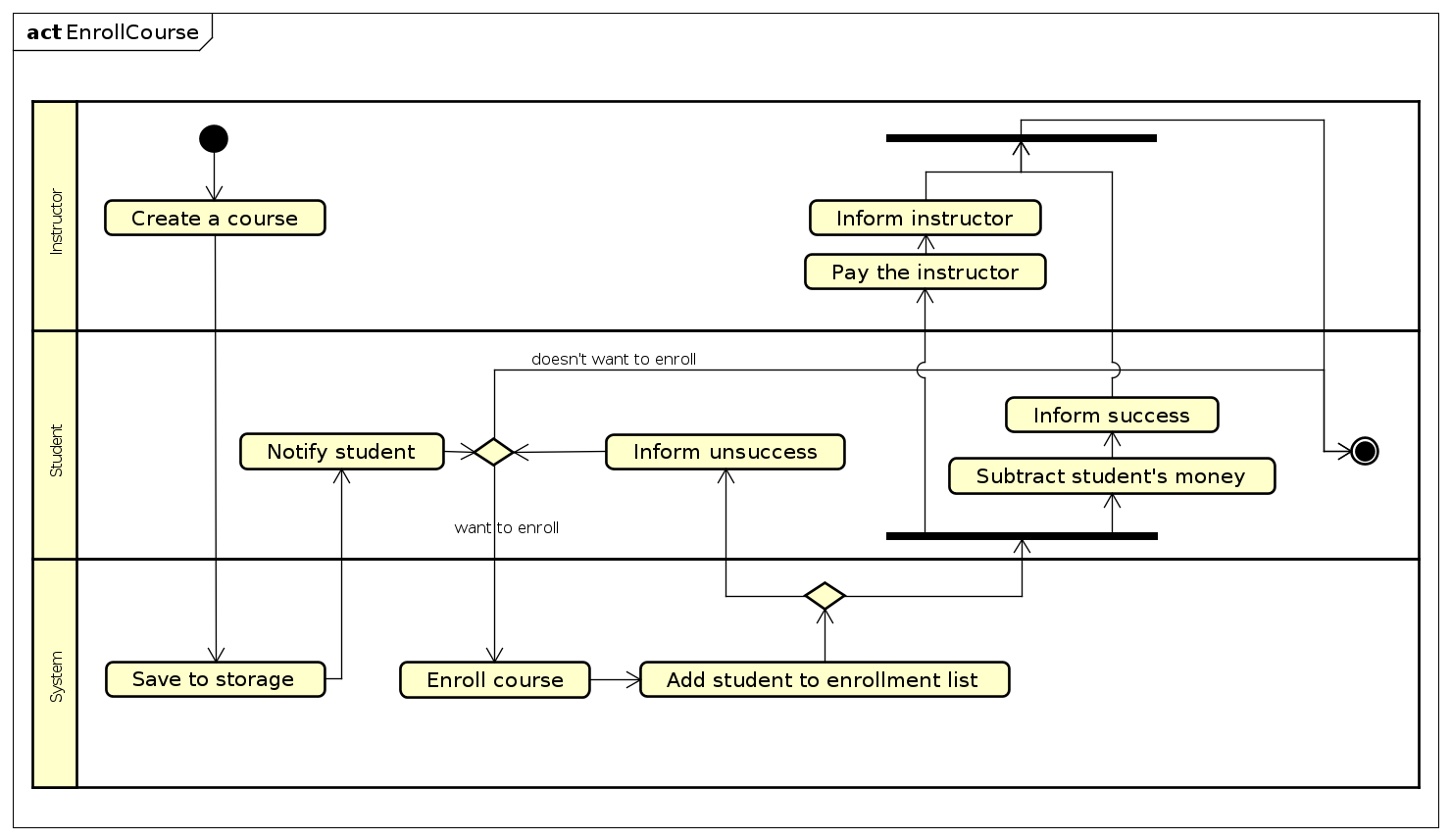
* + - 1. Biểu đồ use case phân rã thống kê hệ thống



**Hình 8.** Biểu đồ use case phân rã thống kê hệ thống

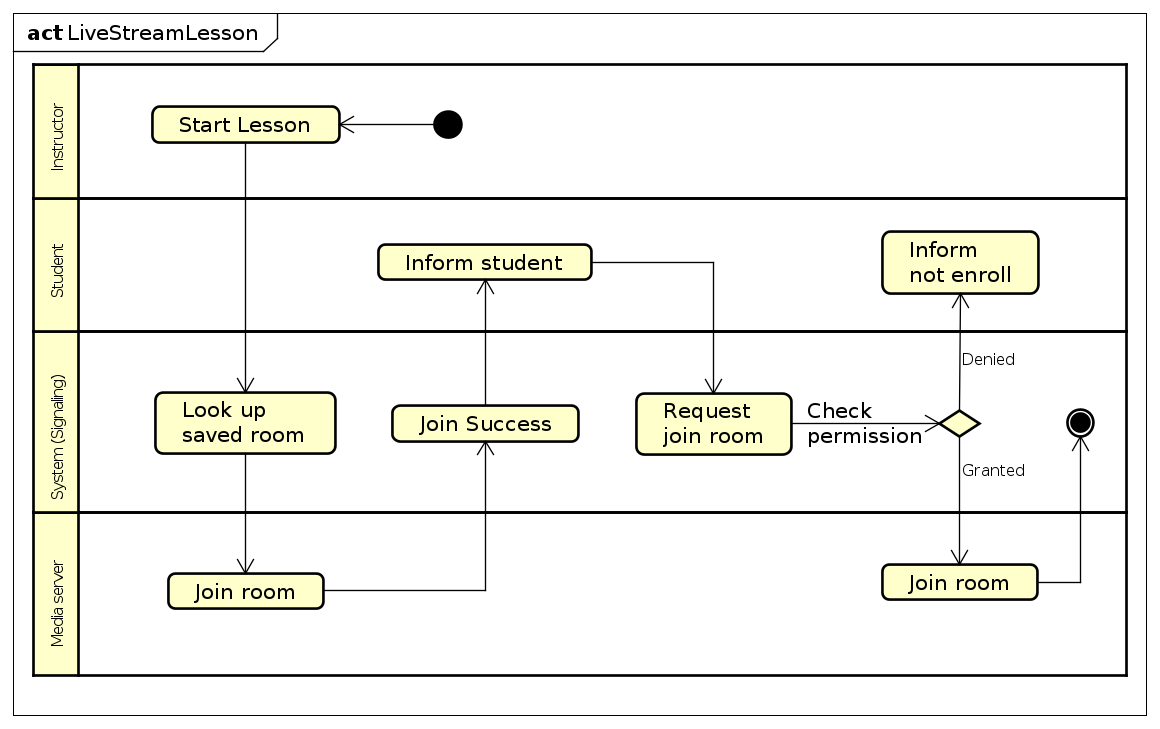
### 1.2.3 Quy trình nghiệp vụ

1.2.3.1 Quy trình nghiệp vụ tạo và tham gia khóa học



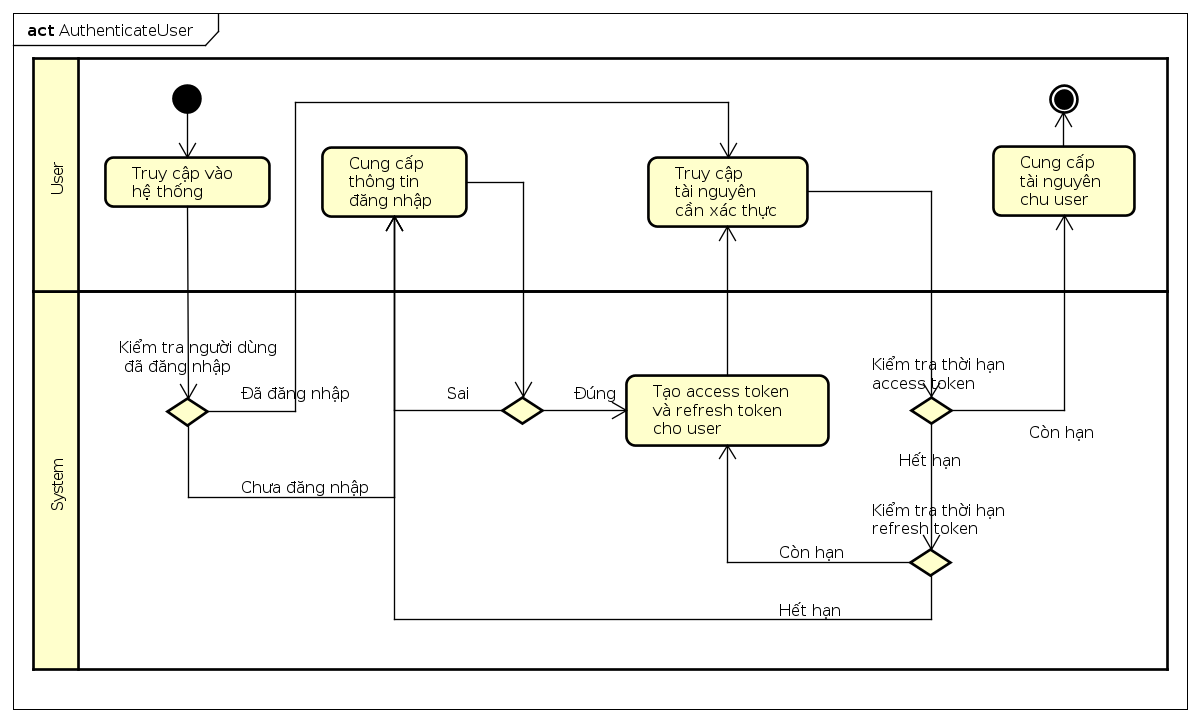
**Hình 5.** Biểu đồ quy trình nghiệp vụ tạo và tham gia khóa học

1.2.3.2 Quy trình nghiệp vụ tham gia phòng học trực tuyến



**Hình 6.** Biểu đồ quy trình nghiệp vụ tham gia phòng học trực tuyến

1.2.3.3 Quy trình nghiệp vụ xác thực người dùng



**Hình 7.** Biểu đồ quy trình nghiệp vụ tạo xác thực người dùng

## Đặc tả chức năng

Sinh viên lựa chọn từ 4 đến 7 use case quan trọng nhất của đồ án để đặc tả chi tiết. Mỗi đặc tả bao gồm ít nhất các thông tin sau: (i) Tên use case, (ii) Luồng sự kiện (chính và phát sinh), (iii) Tiền điều kiện, và (iv) Hậu điều kiện. Sinh viên chỉ vẽ bổ sung biểu đồ hoạt động khi đặc tả use case phức tạp.

### Đặc tả use case OAuth

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Use case** | **UC001** | Tên Use case | **OAuth** |
| **Tác nhân** | Khách (Guest) | | |
| **Tiền điều kiện** | Không | | |
| **Luồng sự kiện chính**  **(Thành công)** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1. | Khách | Yêu cầu liên kết với bên thứ 3 | | 2. | Hệ thống | Trả về url của bên thứ 3 tương ứng | | 3. | Khách | Đăng nhập vào tài khoản bên thứ 3 và cấp quyền cho hệ thống | | 4. | Hệ thống | Kiểm tra thông tin người dùng bên thứ 3 cung cấp | | 5. | Hệ thống | Email bên thứ 3 cung cấp chưa tồn tại trong hệ thống, trả về form điền thêm thông tin cần thiết cho khách | | 6. | Khách | Submit form | | 7. | Hệ thống | Tạo tài khoản mới liên kết với tài khoản bên thứ 3 | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 5.a | Hệ thống | Email đã tồn tại trong hệ thống | | 6.a | Hệ thống | Tài khoản trong hệ thống chưa liên kết với bên thứ 3 | | 7.a | Hệ thống | Request người dùng liên kết tài khoản | | 8.a | Khách | Xác nhận liên kết tài khoản | | 9.a | Hệ thống | Liên kết tài khoản đã tồn tại với tài khoản bên thứ 3 | | 6.a.b | Hệ thống | Tài khoản trong hệ thống đã liên kết | | | |
| **Hậu điều kiện** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1 | Hệ thống | Tạo access token và refresh token cho người dùng | | 2 | Hệ thống | Chuyển về trang trước đó | | | |

\* Dữ liệu đầu vào của form điền thông tin:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc** | **Điều kiện hợp lệ** | **Ví dụ** |
| 1. | username | Tên tài khoản | Có | Duy nhất |  |
| 2. | password | Mật khẩu dùng để đăng nhập | Không | Phải lớn hơn 7 ký tự, chứa cả chữ hoa, thường, số và ký tự đặc biệt | Aa@123123 |
| 3. | rePassword | Nhập lại password | Không | Trùng với password đã nhập |  |
| 4. | role | Loại tài khoản muốn đăng ký | Có |  |  |

### 

### Đặc tả use case Đăng ký

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Use case** | **UC002** | Tên Use case | **Đăng ký** |
| **Tác nhân** | Khách (Guest) | | |
| **Tiền điều kiện** | Không | | |
| **Luồng sự kiện chính**  **(Thành công)** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1. | Khách | Yêu cầu form đăng ký | | 2. | Hệ thống | Trả về form đăng ký | | 3. | Khách | Submit form đăng ký | | 4. | Hệ thống | Tạo tài khoản mới | | 5. | Hệ thống | Gửi email xác thực vào email đăng ký | | 6. | Khách | Chọn xác nhận email đăng ký | | 7. | Hệ thống | Xác nhận tài khoản của khách | | 8. | Hệ thống | Chuyển về trang chủ | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1.b | Khách | Đăng ký bằng tài khoản bên thứ 3 | | 2.b | Hệ thống | Thực hiện luồng hoạt động OAuth | | 4.b | Hệ thống | Thông tin đăng ký của người dùng không hợp lệ | | 5.b | Khách | Quay lại luồng sự kiện 3 | | | |
| **Hậu điều kiện** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1 | Hệ thống | Tạo access token cho người dùng | | | |

\* Dữ liệu đầu vào của form đăng ký:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc** | **Điều kiện hợp lệ** | **Ví dụ** |
| 1. | Họ và tên người dùng |  | Có | Ký tự | Nguyễn Văn A |
| 2. | email | email người dùng là duy nhất | Có | Đúng định dạng email  Duy nhất | s.v.o.a.26@gmail.com |
| 3. | username | Tên tài khoản | Có | Duy nhất |  |
| 4. | password | Mật khẩu dùng để đăng nhập | Có | Phải lớn hơn 7 ký tự, chứa cả chữ hoa, thường, số và ký tự đặc biệt | Aa@123123 |
| 5. | rePassword | Nhập lại password | Có | Trùng với password đã nhập |  |
| 6. | role | Loại tài khoản muốn đăng ký | Có |  |  |

### 

### Đặc tả use case Đăng nhập

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Use case** | **UC003** | Tên Use case | **Đăng nhập** |
| **Tác nhân** | Khách (Guest) | | |
| **Tiền điều kiện** | Khách đã đăng ký tài khoản  Chưa đăng nhập vào hệ thống | | |
| **Luồng sự kiện chính**  **(Thành công)** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1. | Khách | Yêu cầu form đăng nhập | | 2. | Hệ thống | Trả về form đăng nhập | | 3. | Khách | Submit form đăng nhập | | 4. | Hệ thống | Xác thực thông tin đăng nhập | | 5. | Hệ thống | Chuyển về trang chủ | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1.b | Khách | Đăng nhập bằng tài khoản bên thứ 3 | | 2.b | Hệ thống | Thực hiện luồng hoạt động OAuth | | 4.b | Hệ thống | Thông tin đăng ký của người dùng không hợp lệ | | 5.b | Khách | Quay lại luồng sự kiện 3 | | | |
| **Hậu điều kiện** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1 | Hệ thống | Tạo access token cho người dùng | | | |

\* Dữ liệu đầu vào của form đăng ký:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc** | **Điều kiện hợp lệ** | **Ví dụ** |
| 2. | thông tin đăng nhập | email hoặc username của người dùng | Có |  | [s.v.o.a.26@gmail.com](mailto:s.v.o.a.26@gmail.com)/  UserName |
| 4. | password | Mật khẩu dùng để đăng nhập | Có | Phải lớn hơn 7 ký tự, chứa cả chữ hoa, thường, số và ký tự đặc biệt | Aa@123123 |

### 

### Đặc tả use case Tham gia phòng học trực tuyến

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Use case** | **UC004** | Tên Use case | **Tham gia phòng học trực tuyến** |
| **Tác nhân** | Học sinh (Student) | | |
| **Tiền điều kiện** | Học sinh đã mua khóa học | | |
| **Luồng sự kiện chính**  **(Thành công)** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 1. | Học sinh | Yêu cầu tham gia phòng học | | 2. | Hệ thống | Kiểm tra học sinh đã mua khóa học | | 3. | Hệ thống | Lấy thông tin kết nối của phòng học đã lưu trong cơ sở dữ liệu | | 4. | Hệ thống | Kiểm tra lịch học của bài học | | 5. | Hệ thống | Thiết lập kết nối luồng stream video | | 6. | Hệ thống | Thiết lập kết nối kênh chat | | 7. | Hệ thống | Hiển thị giao diện cho người dùng | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | **STT** | **Thực hiện bởi** | **Hành động** | | 3.a | Hệ thống | Học sinh chưa mua khóa học, thông báo và kết thúc luồng sự kiện | | 5.a | Hệ thống | Không phải thời gian học bài học (chưa đến hoặc đã quá thời gian), thông báo cho người dùng | | | |
| **Hậu điều kiện** | Không | | |

## 

## 1.3 Yêu cầu phi chức năng

### 1.3.1 Yêu cầu về bảo mật

Hệ thống có cơ chế xác thực danh tính và phân quyền người dùng theo vai trò. Cơ chế bảo mật đảm bảo việc các tài nguyên hạn chế không thể bị sửa đổi nếu không có quyền hạn chính xác.

### 1.3.2 Yêu cầu về giao diện

Giao diện responsive, thân thiện với người dùng, có thể hoạt động tốt trên nhiều thiết bị và nền tảng. Hệ thống được thiết kế frontend tách biệt với backend, giao tiếp bằng REST API và websocket, giúp dễ dàng xây dựng frontend trên nhiều nền tảng (web, mobile, desktop app ...)

## 1.3.3 Yêu cầu về kỹ thuật Hệ thống được xây dựng với cấu trúc dễ dàng chuyển đổi và mở rộng theo chiều ngang (microservice). Các service có sự phân hóa và giao tiếp với nhau qua interface chung, đảm bảo tính chất open/close, dễ dàng mở rộng, thay đổi các service riêng biệt.

# Công nghệ sử dụng

## 2.1 Frontend

### **2.1.1 React**

React là một thư viện javascript được xây dựng bởi Facebook, thích hợp để xây dựng UI cho các single-page application (SPA).

Các trang web viết bằng React có khả năng lấy dữ liệu từ server mà không cần tải lại trang giúp tăng trải nghiệm người dùng.

## 2.2 Backend

### **2.2.1 Laravel**

Laravel là một PHP framework, có mã nguồn mở và miễn phí, được xây dựng nhằm hỗ trợ phát triển các phần mềm, ứng dụng, theo kiến trúc MVC.

Một số tính năng của Laravel là hệ thống package module composer giúp quản lý các thư viện dễ dàng, nhiều cách truy cập cơ sở dữ liệu quan hệ đa dạng và tiện lợi, các tiện ích hỗ trợ triển khai và bảo trì ứng dụng, cấu trúc và cú pháp có nguyên tắc thống nhất. Hiện nay, Laravel đang là PHP framework phổ biến nhất và tốt nhất.

Đây là framework chính để phát triển và xây dựng những nghiệp vụ cơ bản của hệ thống, cung cấp API cho Frontend.

### **2.2.2 Flask**

Flask là một Python micro web framework, giúp xây dựng hệ thống API nhanh chóng và đơn giản. Flask không yêu cầu tool hay thư viện cụ thể nào nên có thể dễ dàng tích hợp mọi thư viện của python.

Trong hệ thống này, Flask được sử dụng để cung cấp API và train model recommendation system cho hệ thống.

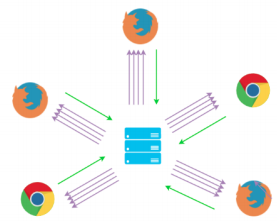
### **2.2.3 Websocket**

Websocket là giao thức hỗ trợ giao tiếp hai chiều giữa client và server để tạo một kết nối trao đổi dữ liệu. Websocket cung cấp các tính năng realtime như thông báo, bình luận, nhắn tin.

### 2.2.4 WebRTC

WebRTC là các API viết bằng javascript giúp giao tiếp theo thời gian thực mà không cần cài plugin hay phần mềm hỗ trợ. WebRTC có khả năng hỗ trợ trình duyệt giao tiếp thời gian thực thông qua Video Call, Voice Call hay transfer data P2P(peer-to-peer), không cần đến plugin, phần mềm khác.

Tuy nhiên trong cài đặt của hệ thống này, các client browser không giao tiếp P2P với nhau mà thông qua một SFU Mediaserver giúp giảm tải cho browser của người dùng và dễ dàng quản lý.



**Hình 8.** Mô hình SFU Media Server

### 2.2.5 Redis

Redis (REmote DIctionary Server) là một mã nguồn mở được dùng để lưu trữ dữ liệu có cấu trúc, có thể sử dụng như một database, bộ nhớ cache hay một message broker.

Trong hệ thống, Redis được sử dụng như một bộ nhớ cache vì dữ liệu của Redis được lưu trên RAM nên tốc độ đọc ghi rất nhanh. Redis cũng đóng vai trò queue để xử lý lần lượt các job không đồng bộ mà các thành phần khác sinh ra.

### 2.2.6 Elasticsearch

Elasticsearch là một search engine (công cụ tìm kiếm) hoạt động như một web server, cung cấp API cho chức năng tìm kiếm.

### 2.2.7 Recommendation systems

Recommendation systems (Hệ thống gợi ý) là một dạng của hệ hỗ trợ ra quyết định, cung cấp giải pháp mang tính cá nhân hóa mà không phải trải qua quá trình tìm kiếm phức tạp. Hệ gợi ý học từ người dùng và gợi ý các sản phẩm tốt nhất trong số các sản phẩm phù hợp.

Các sản phẩm được gợi ý dựa trên số lượng sản phẩm đó đã được bán, dựa trên các thông tin cá nhân của người sử dụng, dựa trên sự phân tích hành vi mua hàng trước đó của người sử dụng và những người sử dụng khác để đưa ra các dự đoán về hành vi mua hàng trong tương lai của khách hàng đó.

Các phương pháp gợi ý:

1. **Content-based**: Đánh giá đặc tính của sản phẩm được recommended. Ví dụ: một người dùng xem rất nhiều các bộ phim về cảnh sát hình sự, vậy thì gợi ý một bộ phim trong cơ sở dữ liệu có chung đặc tính hình sự tới người sử dụng này. Cách tiếp cận này yêu cầu việc sắp xếp các sản phẩm vào từng nhóm hoặc đi tìm các đặc trưng của từng sản phẩm. Tuy nhiên, có những items không có nhóm cụ thể và việc xác định nhóm hoặc đặc trưng của từng sản phẩm đôi khi là bất khả thi.
2. **Collaborative Filtering**: Hệ thống gợi ý sản phẩm dựa trên sự tương quan (similarity) giữa các người dùng và/hoặc sản phẩm. Có thể hiểu rằng ở nhóm này một sản phẩm được recommended tới một người dùng dựa trên những người sử dụng có hành vi tương tự. Ví dụ: người dùng A, B, C đều thích các bài hát của Noo Phước Thịnh. Ngoài ra, hệ thống biết rằng người dùng B, C cũng thích các bài hát của Bích Phương nhưng chưa có thông tin về việc liệu người dùng A có thích Bích Phương hay không. Dựa trên thông tin của những người sử dụng tương tự là B và C, hệ thống có thể dự đoán rằng A cũng thích Bích Phương và gợi ý các bài hát của ca sĩ này tới A.

Collaborative Filtering (CF) có các loại chính:

* Memory-based: Hướng tiếp cận này sử dụng dữ liệu rating của người dùng để tính toán độ tương tự giữa những người sử dụng hoặc giữa những sản phẩm. Ví dụ điển hình của hướng tiếp cận này là neighbourhood-based CF và item-based/user-based top-N recommendations. Ví dụ trong hướng tiếp cận user-based, giá trị rating dự đoán của người sử dụng *u* cho sản phẩm *i* là tổng hợp dữ liệu rating đã có đối với sản phẩm *i* của một số người dùng tương tự *u.*
* Model-based: Hướng tiếp cận này sử dụng models được xây dựng bằng nhiều giải thuật data mining, machine learning khác nhau để dự đoán rating của sản phẩm chưa được rate. Có rất nhiều giải thuật model-based: Bayesian networks, clustering models, singular value decomposition, Markov decision process based models...

Hệ thống sử dụng hướng tiếp cập memory-based, cụ thể hơn là user-based. Bởi vì:

* Content-based: Content-based làm cho các gợi ý có tính chất giống nhau và giống với những sản phẩm mà người dùng đã sử dụng. Điều này không phù hợp với hệ thống cung cấp khóa học này vì các khóa học được gợi ý sẽ lặp lại nội dung gần tương tự với các khóa học mà người dùng đã tham gia.
* Model-based: Model-based cần khối lượng tính toán lớn để xây dựng model cho hệ gợi ý. Những hệ thống có thông tin được cập nhật, làm mới nhiều sẽ gây ra tính toán dư thừa, giảm hiệu năng của hệ thống.
* User-based: User-base lọc cộng tác sẽ gợi ý những sản phẩm dựa trên xu hướng của những người dùng tương tự, giúp gợi ý cho người dùng các khóa học theo trend trong nhóm đối tượng người dùng giống nhau.

## 2.3 Server

### **2.3.1 Amazon web service**

Amazon web service (AWS) là dịch vụ điện toán đám mây cung cấp rất nhiều tính năng cho việc deploy (triển khai) hệ thống

* Amazon Virtual Private Cloud (Amazon VPC): giúp tạo ra một môi trường tách biệt, nơi có thể triển khai hệ thống trong một hệ thống mạng ảo riêng tự định nghĩa. Nhà phát triển có toàn quyền quyết định môi trường mạng ảo này sẽ như thế nào, bao gồm lựa chọn dải IP riêng, tạo mạng con (subnet), bảng định tuyến (route table) và cổng kết nối mạng (net gateway), cũng có thể sử dụng cả IPv4 và IPv6 cho bảo mật và dễ dàng truy cập ứng dụng, tài nguyên.
* Amazon Elastic Compute Cloud (Amazon EC2): là một cơ sở hạ tầng điện toán đám mây được cung cấp bởi AWS giúp cung cấp tài nguyên máy tính ảo hoá theo yêu cầu.
* Amazon Relational Database Service (Amazon RDS): là dịch vụ đám mây do AWS phát triển với mục tiêu cung cấp giải pháp cài đặt, vận hành và mở rộng dành cho cơ sở dữ liệu có quan hệ.
* Amazon Simple Storage Service (Amazon S3): là một kho lưu trữ cloud. Nó được thiết kế để lưu trữ dung lượng lớn, chi phí thấp và có server trên khắp thế giới. S3 cung cấp cho nhà phát triển kho lưu trữ bảo mật, tin cậy và khả năng mở rộng cao.

### **2.3.2 Docker**

Docker là một nền tảng để cung cấp cách để building, deploying và running ứng dụng dễ dàng hơn bằng cách sử dụng các containers (trên nền tảng ảo hóa).

Các containers cho phép lập trình viên đóng gói một ứng dụng với tất cả các phần cần thiết, chẳng hạn như môi trường, thư viện, và gói tất cả ra dưới dạng một package. Bằng cách đó, ứng dụng sẽ chạy trên mọi máy Linux khác bất kể mọi cài đặt tùy chỉnh mà máy có thể có khác với máy được sử dụng để viết code.

Các phần của hệ thống chạy trên AWS EC2 đều được đóng gói trong Docker container, giúp rút ngắn thời gian và tiết kiệm công sức để cài đặt môi trường deploy.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

3.1.1 Kiến trúc phần mềm

**Kiến trúc MVC**: MVC là từ viết tắt bởi 3 từ Model – View – Controller. Đây là mô hình thiết kế sử dụng trong kỹ thuật phần mềm. Mô hình source code thành 3 phần, tương ứng mỗi từ. Mỗi từ tương ứng với một hoạt động tách biệt trong một mô hình.

* Model (M): Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng. Bộ phận này là một cầu nối giữa 2 thành phần bên dưới là View và Controller. Model thể hiện dưới hình thức là một cơ sở dữ liệu hoặc có khi chỉ đơn giản là một file XML bình thường. Model thể hiện rõ các thao tác với cơ sở dữ liệu như cho phép xem, truy xuất, xử lý dữ liệu,…
* View (V): Đây là phần giao diện dành cho người sử dụng. Nơi mà người dùng có thể lấy được thông tin dữ liệu của MVC thông qua các thao tác truy vấn như tìm kiếm hoặc sử dụng thông qua các website.

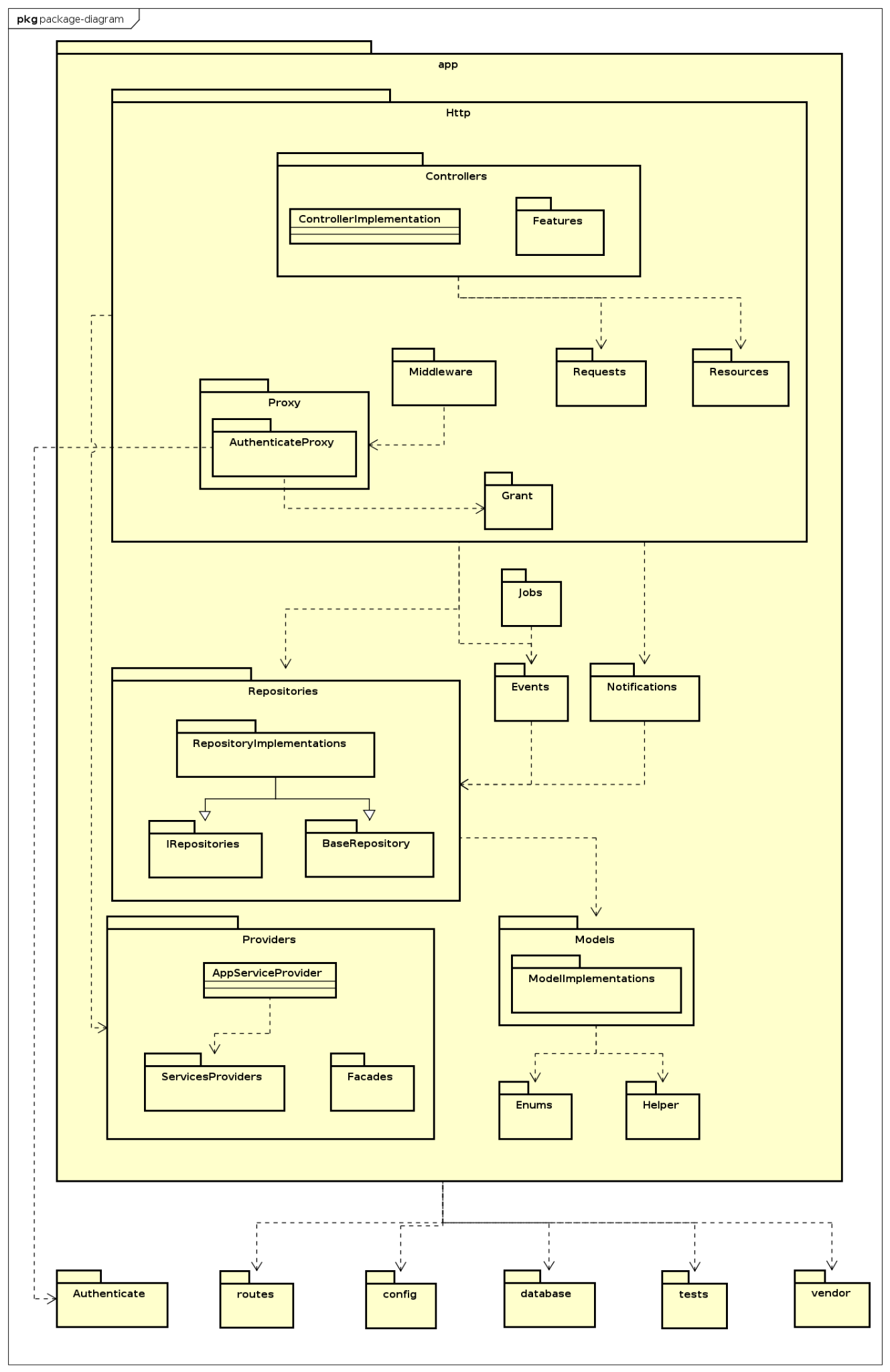
View có chức năng ghi nhận hoạt động của người dùng để tương tác với Controller. Tuy nhiên, View không có mối quan hệ trực tiếp với Controller, cũng không được lấy dữ liệu từ Controller mà chỉ hiển thị yêu cầu chuyển cho Controller. Thông thường, các ứng dụng web sử dụng MVC View như một phần của hệ thống, nơi các thành phần HTML được tạo ra. Tuy nhiên trong hệ thống này, View là một phần tách rời giao tiếp với controller qua API.

* Controller (C): Bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua view. Từ đó, C đưa ra dữ liệu phù hợp với người dùng. Bên cạnh đó, Controller còn có chức năng kết nối với model.

**Repository pattern**: Repository Pattern là lớp trung gian giữa tầng Business Logic (C) và Data Access (M), giúp cho việc truy cập dữ liệu chặt chẽ và bảo mật hơn. Thông thường thì các phần truy xuất, giao tiếp với database nằm rải rác ở trong code, khi bạn muốn thực hiện một thao tác lên database thì phải tìm trong code cũng như tìm các thuộc tính trong bảng để xử lý. Điều này gây lãng phí thời gian và công sức rất nhiều. Với Repository design pattern, thì việc thay đổi ở code sẽ không ảnh hưởng quá nhiều công sức chúng ra chỉnh sửa.

**Redux store**

### 3.1.2 Thiết kế tổng quan



**Hình 8** Biểu đồ phụ thuộc gói

Package Models: Chứa các Class Model - các lớp có thực thể là mapping từ database chứa dữ liệu nghiệp vụ.

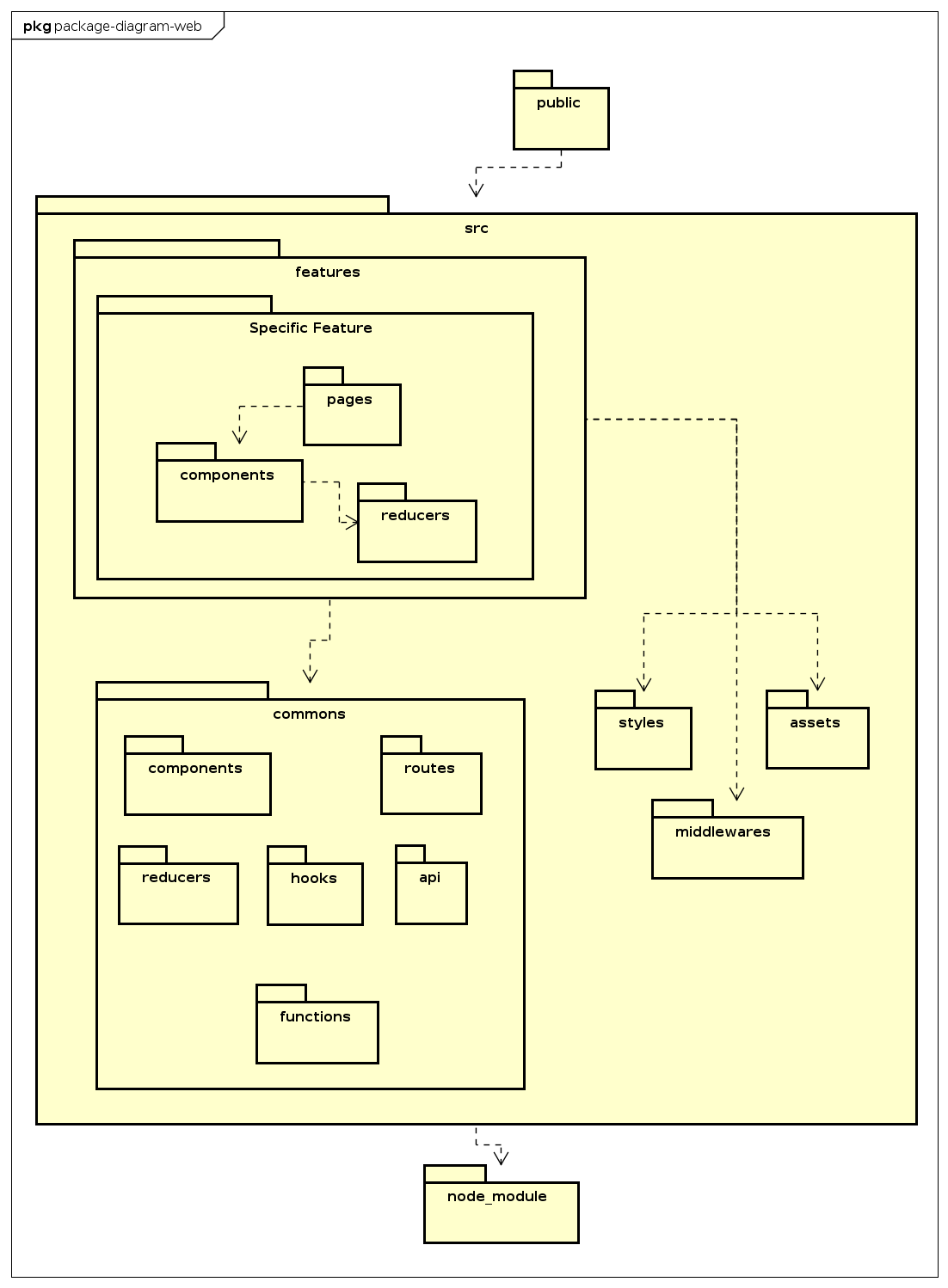
Package Repository: Chứa các Model Repository, có nhiệm vụ thực hiện các logic truy vấn cơ sở dữ liệu. Làm trung gian giữa Controller và Model để tránh phụ thuộc giữa Controller và Model.

Package Service Provider: Đăng ký và khởi tạo sẵn các entity của Service và Facade được sử dụng của hệ thống. Các Service được đăng ký có thể inject vào Controller, tránh gây phụ thuộc giữa các Class, đảm bảo tính loose coupling.

Package Controllers: Chứa các Controller của hệ thống, thực hiện các logic nghiệp vụ. Controller nhận request từ view, xử lý nghiệp vụ với các Model thông qua Repository và lưu trữ, trả lại kết quả xử lý cho controller.

Package Presentation Logic: Giao tiếp với Controller, nhận kết quả và xử lý cho logic hiển thị dữ liệu cho người dùng.

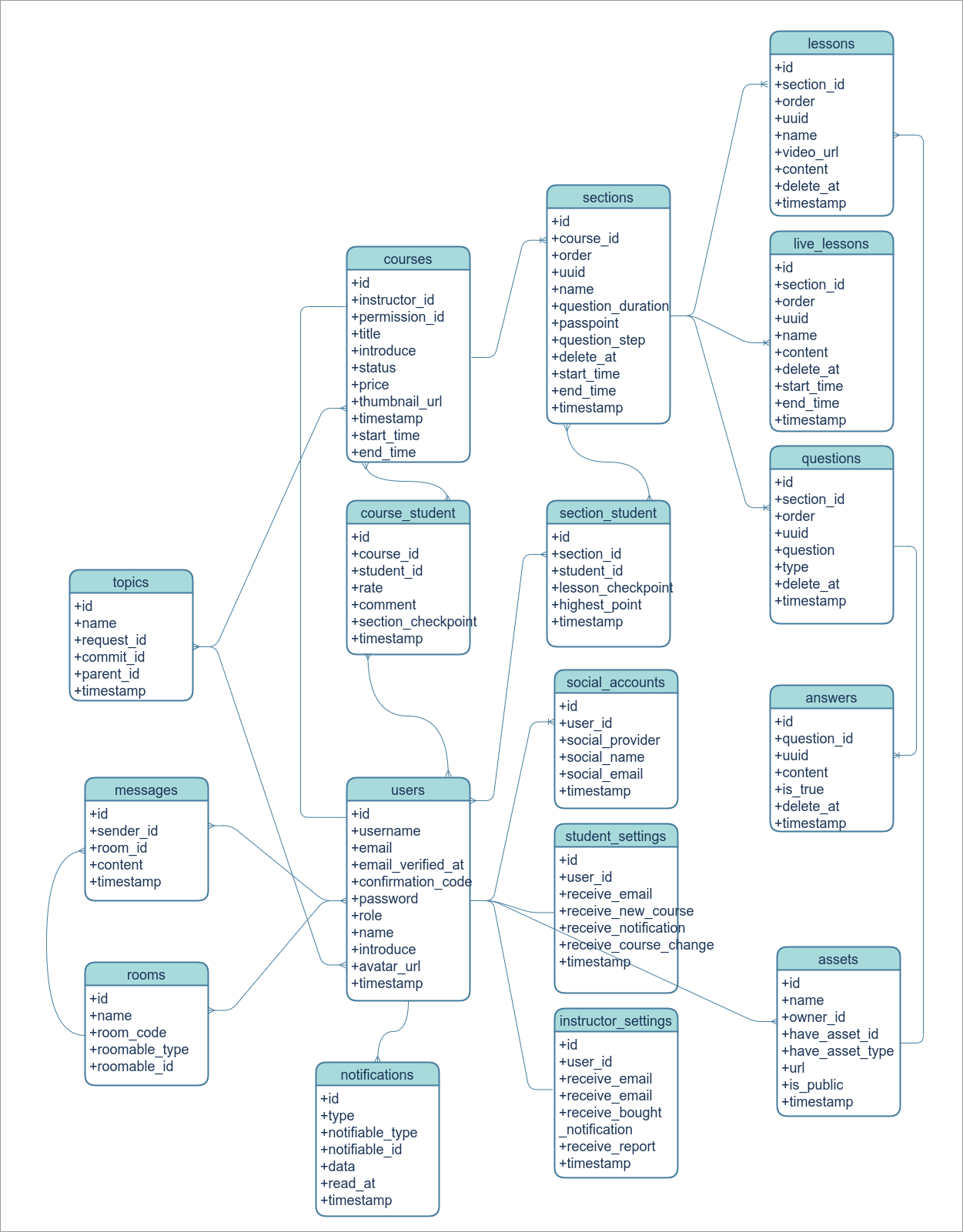
User Interface: Chứa các giao diện hiển thị dữ liệu cho người dùng.



## 3.2 Thiết kế chi tiết

### 3.2.3 Thiết kế cơ sở dữ liệu

### 3.2.3.1 Sơ đồ thực thể liên kết

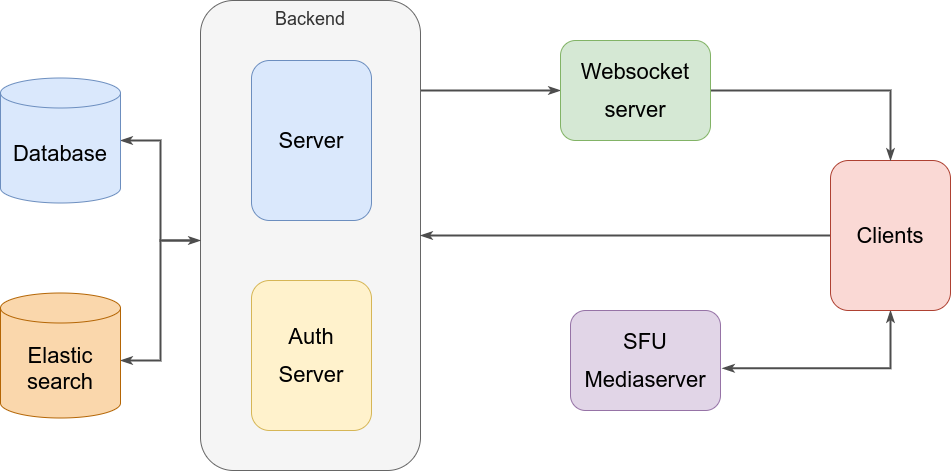


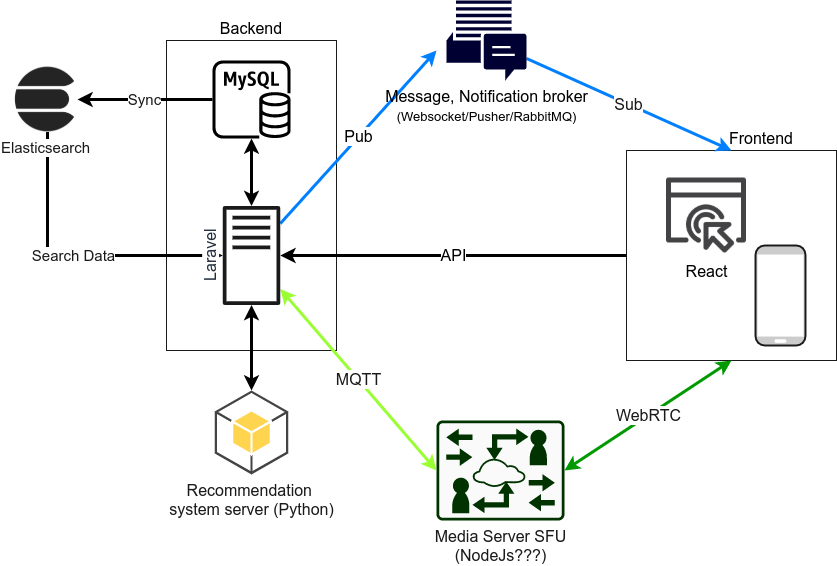
#### 3.2.3.1 Thiết kế chi tiết cơ sở dữ liệu

|  |  |
| --- | --- |
| * Cơ sở dữ liệu users   create table users  (  id bigint unsigned auto\_increment primary key,  name varchar(255) not null,  username varchar(255) null,  email varchar(255) null,  email\_verified\_at timestamp null,  confirmation\_code varchar(255) null,  introduce text null,  avatar\_url varchar(255) null,  password varchar(255) null,  role tinyint default 3 not null,  remember\_token varchar(100) null,  created\_at timestamp null,  updated\_at timestamp null,  constraint users\_email\_unique unique (email),  constraint users\_username\_unique unique (username)  ); | * Cơ sở dữ liệu social\_accounts   create table social\_accounts  (  id bigint unsigned auto\_increment primary key,  user\_id bigint unsigned null,  social\_id varchar(255) not null,  social\_provider varchar(255) not null,  social\_name varchar(255) not null,  social\_email varchar(255) not null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu student\_settings   create table student\_settings  (  user\_id int not null primary key,  receive\_email varchar(255) default '1' null,  receive\_flower\_new\_course tinyint(1) default 0 null,  receive\_notification tinyint(1) default 0 null,  receive\_course\_change tinyint(1) default 0 null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu instructor\_settings   create table instructor\_settings  (  user\_id int not null primary key,  receive\_email varchar(255) null,  receive\_bought\_notification tinyint(1) default 0 null,  receive\_report tinyint(1) default 0 null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu notifications   create table notifications  (  id char(36) not null primary key,  type varchar(255) not null,  notifiable\_type varchar(255) not null,  notifiable\_id bigint unsigned not null,  data text not null,  read\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu assets   create table assets  (  id bigint unsigned auto\_increment primary key,  name varchar(255) not null,  owner\_id int not null,  lesson\_id int null,  message\_id int null,  course\_id int null,  url varchar(255) not null,  is\_public tinyint(1) default 1 not null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu messages   create table messages  (  id bigint unsigned auto\_increment primary key,  sender\_id int not null,  room\_id int not null,  content varchar(255) not null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu rooms   create table rooms  (  id bigint unsigned auto\_increment primary key,  name varchar(255) null,  room\_code varchar(255) null,  roomable\_type varchar(255) null,  roomable\_id int null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu room\_user   create table room\_user  (  room\_id int not null,  user\_id int not null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu user\_topic   create table user\_topic  (  user\_id int not null,  course\_id int not null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu topics   create table topics  (  id bigint unsigned auto\_increment primary key,  name varchar(255) not null,  request\_id int null,  commit\_id int null,  parent\_id int null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu course\_topic   create table course\_topic  (  course\_id int not null,  topic\_id int not null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu courses   create table courses  (  id bigint unsigned auto\_increment primary key,  instructor\_id int not null,  status int default 0 null,  title varchar(255) null,  thumbnail\_url varchar(255) null,  introduce text null,  price double default 0 null,  start\_time datetime null,  end\_time datetime null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu sections   create table sections  (  id bigint unsigned auto\_increment primary key,  course\_id int not null,  `order` int null,  uuid varchar(255) null,  name varchar(255) null,  start\_time datetime null,  end\_time datetime null,  question\_duration int null,  pass\_point double(8, 2) null,  question\_step int null,  deleted\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu lessons   create table lessons  (  id bigint unsigned auto\_increment primary key,  section\_id int not null,  name varchar(255) null,  uuid varchar(255) null,  estimate\_time int null,  video\_url varchar(255) null,  `order` int null,  content mediumtext null,  deleted\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu live\_lessons   create table live\_lessons  (  id bigint unsigned auto\_increment primary key,  section\_id int not null,  uuid varchar(255) null,  name varchar(255) null,  start\_time datetime null,  end\_time datetime null,  `order` int null,  deleted\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); |
| * Cơ sở dữ liệu questions   create table questions  (  id bigint unsigned auto\_increment primary key,  section\_id int not null,  uuid varchar(255) null,  question mediumtext null,  type int default 0 null,  `order` int null,  deleted\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); | * Cơ sở dữ liệu answers   create table answers  (  id bigint unsigned auto\_increment primary key,  uuid varchar(255) null,  question\_id int not null,  content text null,  is\_true tinyint(1) default 0 not null,  deleted\_at timestamp null,  created\_at timestamp null,  updated\_at timestamp null  ); |

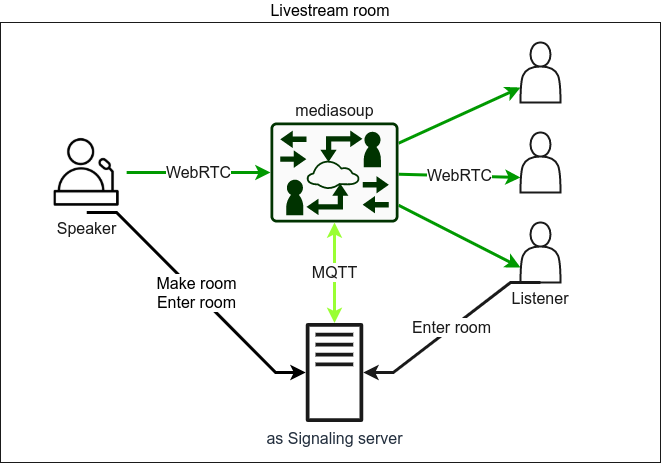
## 3.3 Triển khai

Sinh viên trình bày mô hình và/hoặc cách thức triển khai thử nghiệm/thực tế. Ứng dụng của sinh viên được triển khai trên server/thiết bị gì, cấu hình như thế nào. Kết quả triển khai thử nghiệm nếu có (số lượng người dùng, số lượng truy cập, thời gian phản hồi, phản hồi người dùng, khả năng chịu tải, các thống kê, v.v.)

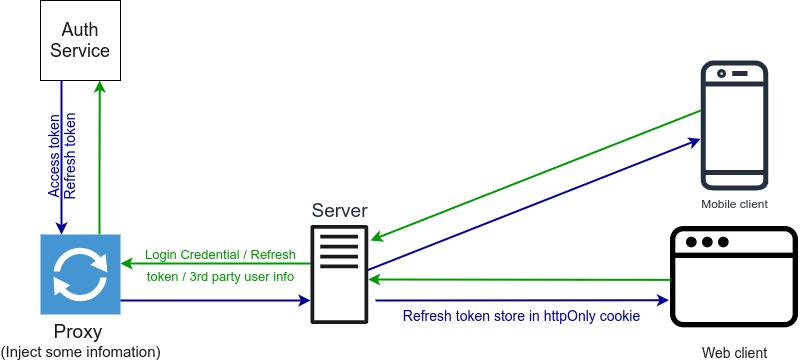




**Hình 5** Sơ đồ tổng quan các công nghệ sử dụng



**Hình 6** Sơ đồ công nghệ sử dụng cho phòng học trực tuyến



**Hình 7** Sơ đồ cơ chế xác thực người dùng