

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



BÁO CÁO BÀI TẬP LỚN
KIẾN TRÚC PHẦN MỀM
ĐỀ TÀI:
HỆ THỐNG GIA SƯ THÔNG MINH (ITS)

Giảng viên hướng dẫn: Trần Trương Tuấn Phát

Sinh viên thực hiện:

Nguyễn Anh Khoa	2211614
Phạm Quang Minh	2212075
Đoàn Ngọc Hoàng Sơn	2212935
Nguyễn Văn Sơn	2212949
Trần Nam Sơn	2212956
Nguyễn Hiệp Tài	2212985
Huỳnh Cảnh Thịnh	2213272

TP. HỒ CHÍ MINH, THÁNG 12 NĂM 2025

Mục lục

1	Giới thiệu	1
1.1	Đặt vấn đề	1
1.2	Mô tả ý tưởng	2
2	Cơ sở lý thuyết và công nghệ	3
2.1	Công nghệ Lỗi và Backend	3
2.2	Công nghệ Frontend	3
2.3	Nền tảng Cloud: Amazon Web Services (AWS)	4
2.4	Docker và Amazon Elastic Container Service (ECS)	4
2.5	Spring Cloud Gateway	5
2.6	Spring Cloud Discovery và Eureka	5
2.6.1	Eureka Server – Service Registry	5
2.6.2	Eureka Client – Service Discovery	6
2.7	Giao tiếp giữa các Service: REST API trên mạng nội bộ ECS	6
2.8	AWS Amplify	6
3	Yêu cầu về hệ thống	7
3.1	Xác định người dùng	7
3.2	Yêu cầu chức năng	7
3.3	Yêu cầu phi chức năng	8
4	Đặc tả chi tiết các use-case	10
4.1	Quản lý hệ thống	10
4.1.1	Tổng quan	10
4.1.2	Đặc tả chi tiết	10
4.2	Quản lý tài khoản	12
4.2.1	Tổng quan	12
4.2.2	Đặc tả chi tiết	12
4.3	Quản lý quyền truy cập	15
4.3.1	Tổng quan	15
4.3.2	Đặc tả chi tiết	15
4.4	Quản lý khóa học	18
4.4.1	Tổng quan	18
4.4.2	Đặc tả chi tiết	18
4.5	Học khóa học	20
4.5.1	Tổng quan	20
4.5.2	Đặc tả chi tiết	20
5	Phân tích & Thiết kế hệ thống	24
5.1	Entity Relationship Diagram (ERD)	24
5.1.1	Tổng quan	24
5.1.2	Mapping	27
5.2	Kiến trúc hệ thống	27
5.2.1	Đặc tính kiến trúc (Architecture characteristics)	27
5.2.2	So sánh các phong cách kiến trúc khả thi	28
5.2.3	Thiết kế kiến trúc	29
5.2.3.a	Module view	29
5.2.3.b	Component & Connector View	32
5.2.3.c	Allocation view	33
6	Hiện thực	35
7	Triển khai	36
8	Tổng kết	37
8.1	Nhận xét	37
8.2	Hướng phát triển	37

9	Phụ lục	38
9.1	ADR	38
9.1.1	ADR 00: Lựa chọn kiến trúc ban đầu: Microservices Architecture (MSA)	38
9.1.2	ADR 01: Chuyển đổi kiến trúc từ Microservices (MSA) sang Service-Based Architecture (SBA)	38
10	Tài liệu tham khảo	39
11	Phụ lục	40

Danh sách hình vẽ

1	Tam giác học tập	1
2	Các công nghệ Back-end	3
3	React	3
4	Amazon Web Services	4
5	Docker và Amazon Elastic Container Service	4
6	AWS Amplify	6
7	Tổng quan các Use-case	10
8	Các use-case về Quản lý hệ thống	10
9	Các use-case về Quản lý tài khoản	12
10	Các use-case về Quản lý quyền chỉnh sửa khóa học	15
11	Các use-case về Quản lý khóa học	18
12	Các use-case về Học khóa học	20
13	Entity Relationship Diagram	24
14	Relational Mapping Diagram	27
15	Module view - Nguyên tắc tổ chức	29
16	Module view - Package diagram	30
17	Component & Connector view	32
18	Component & Connector view	33

Danh sách bảng

1	Bảng User Stories	8
2	Danh sách yêu cầu phi chức năng (NFR)	9
3	Usecase «Cấu hình hệ thống»	10
4	Usecase «Xem nhật ký hoạt động AI»	11
5	Usecase «Xem nhật ký kiểm toán (Audit log)»	11
6	Usecase «Tạo hoặc Xóa tài khoản»	12
7	Usecase «Đặt lại mật khẩu (Reset password)»	13
8	Usecase «Đăng ký tài khoản»	13
9	Usecase «Đăng nhập»	14
10	Usecase «Yêu cầu đặt lại mật khẩu»	14
11	Usecase «Cấp và thu hồi quyền Trưởng khoa»	15
12	Usecase «Phê duyệt yêu cầu quyền truy cập»	16
13	Usecase «Thu hồi quyền truy cập»	16
14	Usecase «Yêu cầu quyền truy cập»	17
15	Usecase «Đề xuất chỉnh sửa khóa học»	18
16	Usecase «Tạo khóa học»	18
17	Usecase «Sửa nội dung khóa học»	19
18	Usecase «Ghi danh khóa học»	19
19	Usecase «Xem đề cương khóa học»	20
20	Usecase «Xem tài liệu học tập»	21
21	Usecase «Xem phản hồi thống kê»	21
22	Usecase «Làm bài kiểm tra (quiz)»	22
23	Usecase «Tạo báo cáo thống kê»	22
24	Usecase «Tham gia diễn đàn khóa học»	23
26	So sánh giữa Service-Based Architecture (SBA) và Microservices Architecture (MSA)	28

1 Giới thiệu

1.1 Đặt vấn đề

Trong môi trường giáo dục truyền thống, việc giảng dạy chủ yếu diễn ra theo phương pháp đồng bộ, nơi giáo viên truyền đạt kiến thức chung cho cả lớp. Điều này tạo ra một số thách thức lớn:

Giới hạn trong học tập cá nhân hóa

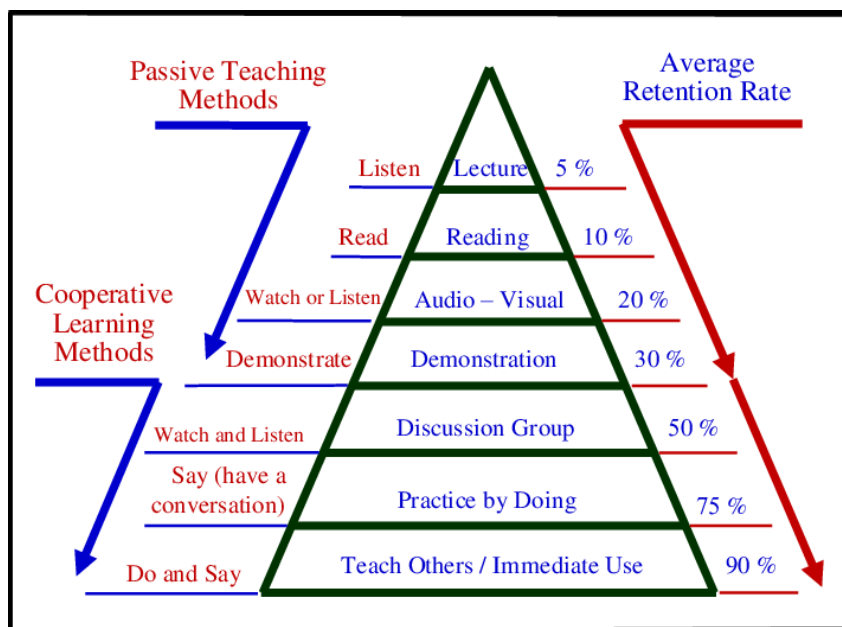
Việc giảng dạy truyền thống khiến giáo viên gặp khó khăn trong việc đáp ứng nhu cầu, tốc độ và phong cách học tập đa dạng của từng học viên.

- **Tốc độ và nhu cầu:** Mỗi học viên có tốc độ học khác nhau, nhưng thời gian và nguồn lực hạn chế khiến việc cung cấp trải nghiệm học tập cá nhân hóa, phản hồi tức thì và hỗ trợ thích ứng theo thời gian thực trở thành một thách thức lớn.
- **Thiếu hiệu quả:** Điều này dẫn đến sự thiếu hiệu quả trong việc phát huy tối đa tiềm năng của mỗi người học.

Sự thống trị của phương pháp giảng dạy thụ động

Các phương pháp giảng dạy truyền thống, nơi thông tin chảy một chiều từ người hướng dẫn đến học viên, thường dựa trên các hình thức thụ động.

- **Tỷ lệ lưu giữ kiến thức thấp:** Theo nguyên lý của Tam giác Học tập (Learning Pyramid)¹, các phương pháp thụ động như Listen hay Read có tỷ lệ lưu giữ trung bình rất thấp (chỉ 5% - 10%).



Hình 1: Tam giác học tập

- **Thiếu kỹ năng mềm:** Các phương pháp này không tạo điều kiện cho học viên tham gia vào các hoạt động học tập chủ động như giải quyết vấn đề, làm việc nhóm, hoặc giao tiếp, vốn là những kỹ năng mềm thiết yếu mà các chương trình kỹ thuật cần phải nâng cao.

Hạn chế trong đánh giá và theo dõi tiến độ học tập

Hệ thống giáo dục truyền thống chủ yếu dựa vào các bài kiểm tra định kỳ như giữa kỳ và cuối kỳ, dẫn đến việc đánh giá không phản ánh đầy đủ quá trình học tập của học viên.

- **Thiếu dữ liệu thời gian thực:** Giáo viên không thể theo dõi tiến độ của từng học viên theo từng buổi học, dẫn đến khó phát hiện sớm những lỗ hổng kiến thức.
- **Quá trình bị gián đoạn:** Học viên chỉ nhận được phản hồi sau khi hoàn thành bài kiểm tra, nên khó điều chỉnh chiến lược học tập ngay lập tức.
- **Không khuyến khích sự phát triển liên tục:** Khi đánh giá chỉ tập trung vào điểm số, việc hình thành năng lực lâu dài và tư duy phản biện bị xem nhẹ.

¹<https://www.educationcorner.com/the-learning-pyramid/>

Thiếu đa dạng trong nội dung học tập và phương tiện giảng dạy

Phần lớn nội dung trong mô hình truyền thống được thiết kế theo chuẩn chương trình chung, không phân tầng theo trình độ và không linh hoạt theo nhu cầu cá nhân.

- Ít mức độ truy cập: Học viên không có cơ hội lựa chọn nội dung phù hợp với khả năng—ví dụ: nâng cao, bổ sung cơ bản hoặc nội dung mở rộng.
- Thiếu phương tiện tương tác: Việc phụ thuộc vào sách giáo khoa và bài giảng trên lớp hạn chế việc tiếp cận tài liệu multimedia hoặc công cụ mô phỏng trực quan.
- Cản trở mô hình học tập tự định hướng: Học viên khó tiếp cận tài liệu ở mọi lúc, mọi nơi, dẫn đến sự bị động trong việc xây dựng lộ trình học riêng.

1.2 Mô tả ý tưởng

Để giải quyết những hạn chế của giáo dục truyền thống và tối đa hóa hiệu quả học tập, Hệ thống Gia sư Thông minh (ITS) đã được nghiên cứu và phát triển. ITS là một ứng dụng phần mềm mạnh mẽ sử dụng các kỹ thuật Trí tuệ Nhân tạo (AI) để mô phỏng và cung cấp hướng dẫn cá nhân hóa. ITS tự điều chỉnh linh hoạt theo tốc độ, khả năng, điểm mạnh, điểm yếu và sở thích của từng người học thông qua các tính năng tiên tiến, tập trung vào học tập chủ động:

- Tăng cường thực hành: ITS cung cấp bài tập thực hành thiết kế theo thời gian thực, kèm theo phản hồi thích ứng để đảm bảo học viên liên tục áp dụng kiến thức và phát triển kỹ năng giải quyết vấn đề.
- Hỗ trợ kỹ năng mềm: ITS tích hợp các module hỗ trợ thảo luận nhóm, giúp học viên luyện tập giao tiếp và hợp tác.
- Tự đánh giá và giải thích: Để đạt mức độ lưu giữ kiến thức cao nhất, ITS yêu cầu học viên giải thích các khái niệm thiết kế cho gia sư AI, buộc họ phải trở thành "người dạy", từ đó củng cố khả năng tự điều chỉnh (Self-Regulation) và hiểu sâu.
- Lộ trình học tập tùy chỉnh: AI của hệ thống phân tích chi tiết hoạt động học tập của học viên để tự động điều chỉnh lộ trình, gợi ý các tài liệu (như video, bài đọc) và chủ đề tiếp theo có độ khó phù hợp, thay thế cho chương trình giảng dạy chung cứng nhắc.
- Đánh giá liên tục: Thay vì chỉ dựa vào kiểm tra định kỳ, ITS cung cấp các bài kiểm tra ngắn sau mỗi nội dung của chương. Hệ thống AI có khả năng tự động tạo ra các câu hỏi đánh giá mới, đa dạng, phân tầng theo độ khó và tập trung vào các lỗ hổng kiến thức cụ thể của từng học viên.
- Phát hiện sớm lỗ hổng kiến thức: Mọi tương tác của học viên đều được ghi lại và phân tích chi tiết. Điều này giúp phát hiện sớm các lĩnh vực mà học viên đang gặp khó khăn, cho phép can thiệp và hỗ trợ kịp thời, đảm bảo không bỏ sót bất kỳ lỗ hổng kiến thức nào.

Nhờ những tính năng này, ITS cung cấp phản hồi, gợi ý và lộ trình học tập tùy chỉnh, mang lại hiệu quả tương đương với mô hình gia sư một kèm một, giúp người học phát huy tối đa tiềm năng của mình.

2 Cơ sở lý thuyết và công nghệ

Hệ thống được xây dựng trên nền tảng Java/Spring Boot kết hợp với kiến trúc phân tán Spring Cloud và giao diện người dùng React. Việc lựa chọn bộ công nghệ này nhằm tận dụng khả năng phát triển nhanh, tính ổn định cao và sự hỗ trợ mạnh mẽ cho kiến trúc Service-Based Architecture – SBA.

2.1 Công nghệ Lỗi và Backend

Các dịch vụ nghiệp vụ và hỗ trợ (Domain & Infrastructure Services) đều sử dụng hệ sinh thái Spring.

- Spring Boot: Đây là framework phát triển lõi cho toàn bộ các dịch vụ backend, bao gồm Identity Access Management, Quiz, System, Learning, Course, AI, và Service Registry. Spring Boot cung cấp môi trường chạy độc lập và cấu hình tự động (auto-configuration), đảm bảo mỗi module là một đơn vị triển khai rời rạc và dễ dàng quản lý.
- Spring Cloud Gateway: Công nghệ này được sử dụng để triển khai module API Gateway. Nó đóng vai trò là điểm truy cập duy nhất cho client, chịu trách nhiệm chính trong việc định tuyến yêu cầu tới dịch vụ tương ứng và thực hiện lọc yêu cầu (filtering) trước khi đến các dịch vụ nghiệp vụ.
- Spring Cloud Discovery (Eureka Server): Đây là công nghệ nền tảng cho module Service Registry. Nó đóng vai trò là máy chủ đăng ký dịch vụ, lưu trữ metadata và danh sách các instance đang chạy của các dịch vụ.
- Spring Cloud Discovery (Eureka Client): Tất cả các Service nghiệp vụ (IAM, Quiz, System, Learning, Course, AI) và API Gateway đều được cấu hình là Eureka Client. Điều này cho phép chúng tự động đăng ký với Eureka Server và tận dụng cơ chế khám phá dịch vụ (Service Discovery) để xác định instance hợp lệ tại thời điểm chạy (runtime).
- Spring AI: Đây là thư viện được sử dụng trong AI Service. Nó hỗ trợ việc tích hợp các mô hình Trí tuệ Nhân tạo (AI Models) bên ngoài một cách dễ dàng. Spring AI giúp AI Service thực hiện các chức năng như phân tích hành vi và tiến trình học, cũng như đề xuất lộ trình học tập phù hợp dựa trên dữ liệu phân tích.



Hình 2: Các công nghệ Back-end

2.2 Công nghệ Frontend

React là một thư viện JavaScript mã nguồn mở do Facebook phát triển, được sử dụng rộng rãi để xây dựng giao diện người dùng (User Interface – UI), đặc biệt là các ứng dụng web có tính tương tác cao. React cho phép nhà phát triển xây dựng giao diện thông qua các thành phần (components) độc lập, có khả năng tái sử dụng, giúp tổ chức mã nguồn rõ ràng và dễ bảo trì hơn.

Một trong những ưu điểm nổi bật của React là sử dụng Virtual DOM – một cấu trúc DOM ảo giúp tối ưu hóa hiệu năng. Khi giao diện thay đổi, React chỉ cập nhật những phần tử cần thiết thay vì render lại toàn bộ trang, nhờ đó cải thiện tốc độ xử lý và trải nghiệm của người dùng.

React hỗ trợ mô hình lập trình declarative (khai báo), cho phép mô tả giao diện theo trạng thái của ứng dụng. Khi trạng thái thay đổi, React tự động cập nhật UI tương ứng, giảm thiểu lỗi và đơn giản hóa việc quản lý vòng đời giao diện.

Ngoài ra, React có một hệ sinh thái phát triển rộng lớn bao gồm:

- React Hooks: Cung cấp cách tiếp cận mới để quản lý state và vòng đời trong function components.
- React Router: Hỗ trợ điều hướng nhiều trang trong ứng dụng single-page.



Hình 3: React

Nhờ vào cấu trúc linh hoạt, hiệu năng cao và cộng đồng lớn mạnh, React trở thành một trong những công nghệ phổ biến nhất được lựa chọn trong phát triển các ứng dụng web hiện đại, đặc biệt trong các dự án cần giao diện tương tác, trải nghiệm mượt mà và khả năng mở rộng tốt

2.3 Nền tảng Cloud: Amazon Web Services (AWS)

Amazon Web Services (AWS) là nền tảng điện toán đám mây hàng đầu thế giới, cung cấp hơn 200 dịch vụ toàn diện bao phủ hầu hết mọi nhu cầu về hạ tầng và ứng dụng. Với các trung tâm dữ liệu được phân bố trên toàn cầu, AWS mang đến khả năng mở rộng linh hoạt, độ sẵn sàng cao, bảo mật nhiều lớp, cùng với mô hình chi phí tối ưu theo nhu cầu sử dụng.



Hình 4: Amazon Web Services

AWS cung cấp đầy đủ bốn nhóm dịch vụ cốt lõi trong kiến trúc cloud hiện đại:

- Compute: EC2, ECS, Lambda cho phép triển khai ứng dụng dưới nhiều mô hình từ máy ảo, container đến serverless.
- Storage: S3, EBS, EFS hỗ trợ lưu trữ đối tượng, khối và file system với độ bền dữ liệu cao.
- Networking: VPC, Elastic Load Balancing, Route 53 cho phép kiểm soát mạng nội bộ, định tuyến và phân phối tải.
- Database: RDS, DynamoDB, Aurora cung cấp hệ quản trị cơ sở dữ liệu quan hệ và NoSQL hiệu năng cao.

Một điểm mạnh nổi bật của AWS là khả năng tự động mở rộng (auto scaling) và cân bằng tải (load balancing) giúp hệ thống đáp ứng linh hoạt khi lưu lượng tăng đột biến mà không gián đoạn hoạt động. Đồng thời, AWS áp dụng mô hình bảo mật theo nhiều lớp (multi-layer security) gồm mã hoá dữ liệu, kiểm soát truy cập theo IAM, network isolation qua VPC và giám sát liên tục với CloudWatch.

Nhờ các đặc điểm này, việc triển khai hệ thống trên AWS giúp giảm tải phần vận hành hạ tầng, tăng độ tin cậy, đồng thời cho phép nhóm phát triển tập trung vào logic ứng dụng thay vì lo lắng về máy chủ hay cấu hình mạng. Đây là nền tảng phù hợp cho các hệ thống IoT, AI và xử lý dữ liệu theo thời gian thực, đặc biệt khi yêu cầu khả năng mở rộng cao và vận hành ổn định.

2.4 Docker và Amazon Elastic Container Service (ECS)

Để đảm bảo tính nhất quán trong quá trình phát triển và triển khai, toàn bộ các thành phần backend của hệ thống được đóng gói dưới dạng Docker container. Docker cho phép đóng gói mã nguồn kèm theo môi trường chạy (runtime, thư viện, cấu hình) vào một Docker image duy nhất. Nhờ vậy, ứng dụng có thể vận hành giống nhau ở mọi môi trường — từ máy lập trình viên, môi trường staging, cho đến môi trường production. Điều này giúp giảm thiểu xung đột phụ thuộc (dependency conflict), đơn giản hoá quá trình triển khai và tăng tốc độ phát hành phiên bản mới.

Để quản lý và vận hành các container này, hệ thống sử dụng Amazon Elastic Container Service (ECS), một dịch vụ điều phối container (container orchestration) mạnh mẽ của AWS. ECS cho phép triển khai các container lên một ECS Cluster, theo dõi trạng thái của chúng và tự động khởi động lại khi xảy ra lỗi. ECS hỗ trợ hai mô hình chạy: EC2 (tự quản lý EC2 instances) và Fargate (serverless, không cần quản lý máy chủ), giúp linh hoạt tối ưu chi phí và hiệu năng.



Hình 5: Docker và Amazon Elastic Container Service

Thông qua ECS, hệ thống được hưởng các cơ chế vận hành tự động như:

- Tự động mở rộng (Auto Scaling): tăng hoặc giảm số lượng container dựa trên tải.
- Health Check và tự phục hồi: container lỗi sẽ được thay thế ngay lập tức để duy trì tính sẵn sàng.
- Load Balancing: tích hợp với Elastic Load Balancer (ELB) để phân phối đều lưu lượng.
- Triển khai phiên bản mới không gián đoạn: thông qua cơ chế rolling update.

Việc áp dụng Docker và ECS giúp hệ thống đạt được khả năng mở rộng linh hoạt, ổn định trong vận hành, dễ dàng bảo trì và phù hợp với kiến trúc microservices. Đồng thời, nền tảng cloud-native của ECS giúp giảm tối đa công sức quản lý hạ tầng, cho phép nhóm phát triển tập trung hoàn toàn vào logic ứng dụng.

2.5 Spring Cloud Gateway

Spring Cloud Gateway đóng vai trò là tầng định tuyến trung tâm (API Gateway) trong kiến trúc của hệ thống. Đây là một giải pháp hiện đại được xây dựng trên nền tảng Spring WebFlux và mô hình lập trình phản ứng (reactive programming), cho phép xử lý lượng lớn yêu cầu với hiệu năng cao và độ trễ thấp. Spring Cloud Gateway chịu trách nhiệm tiếp nhận toàn bộ lưu lượng từ người dùng hoặc từ API Gateway bên ngoài, sau đó định tuyến chúng đến đúng dịch vụ phía backend trong hệ thống.

Về mặt chức năng, Spring Cloud Gateway cung cấp nhiều khả năng phù hợp cho hệ thống:

- **Định tuyến linh hoạt (Dynamic Routing):** cho phép điều hướng yêu cầu tới các service khác nhau dựa trên URL, header, tham số truy vấn hoặc quy tắc tùy chỉnh.
- **Cân bằng tải (Load Balancing):** tích hợp liền mạch với Spring Cloud LoadBalancer hoặc Service Discovery để phân phối yêu cầu đều giữa các bản sao container.
- **Lọc và xử lý trước/sau (Pre/Post Filters):** hỗ trợ chèn logic xử lý như xác thực, ghi log, giới hạn tốc độ (rate limiting), sửa đổi header hoặc payload trước khi chuyển đến service đích.
- **Hỗ trợ bảo mật:** dễ dàng tích hợp với Spring Security để triển khai cơ chế xác thực, phân quyền, JWT, OAuth2.
- **Khả năng mở rộng:** được thiết kế theo kiến trúc phi đồng bộ, phù hợp cho phân tán tải cao trong môi trường container.

Với vai trò cổng vào duy nhất của hệ thống, Spring Cloud Gateway giúp đơn giản hóa giao tiếp, giảm độ phức tạp giữa client và các microservice, đồng thời tăng tính bảo mật bằng cách che giấu cấu trúc nội bộ. Nhờ hoạt động trên nền phản ứng (reactive), gateway đạt hiệu năng tốt ngay cả khi số lượng kết nối và yêu cầu tăng cao.

2.6 Spring Cloud Discovery và Eureka

Trong hệ thống, việc các dịch vụ có thể tìm thấy và giao tiếp với nhau một cách tự động là một yêu cầu quan trọng. Spring Cloud Discovery cùng với Eureka cung cấp một giải pháp hoàn chỉnh cho cơ chế Service Registry và Service Discovery, đảm bảo sự linh hoạt và khả năng mở rộng của hệ thống phân tán.

2.6.1 Eureka Server – Service Registry

Eureka Server hoạt động như một trung tâm đăng ký dịch vụ (Service Registry). Nó duy trì một danh sách động các service instance đang hoạt động trong hệ thống cùng với metadata liên quan (như địa chỉ, port, trạng thái...). Các dịch vụ khi khởi động sẽ gửi yêu cầu đăng ký (registration) đến Eureka Server và định kỳ gửi heartbeat để thông báo rằng chúng vẫn còn hoạt động.

Nhờ đó, hệ thống có thể:

- **Xóa bỏ cấu hình tĩnh:** Không cần khai báo cứng địa chỉ IP hay endpoint của từng service.
- **Hỗ trợ mở rộng tự động:** Khi có service instance mới xuất hiện hoặc bị hỏng, registry sẽ phản ánh ngay lập tức.
- **Tăng tính linh hoạt:** Cho phép dịch chuyển container, scaling theo tải mà không ảnh hưởng tới cấu trúc hệ thống.

2.6.2 Eureka Client – Service Discovery

Các dịch vụ trong hệ thống đóng vai trò là Eureka Client, nghĩa là chúng sẽ:

- **Tự động đăng ký** với Eureka Server khi khởi động.
- **Lấy danh sách các service khác** từ registry để biết cách giao tiếp mà không cần cấu hình thêm.
- **Cập nhật danh sách instance theo thời gian thực**, bảo đảm rằng mỗi yêu cầu được định tuyến đến service đang khả dụng.

Cơ chế Service Discovery giúp các dịch vụ giao tiếp theo kiểu “tên dịch vụ” thay vì địa chỉ cố định, tạo ra:

- **Khả năng chịu lỗi cao:** Khi một instance ngừng hoạt động, client sẽ tự động chuyển sang instance khác.
- **Tự động cân bằng tải:** Kết hợp với các công cụ như Spring Cloud LoadBalancer hoặc Gateway.
- **Tối ưu hoá vận hành trong môi trường container:** Các địa chỉ container thay đổi liên tục nhưng không ảnh hưởng đến toàn hệ thống.

2.7 Giao tiếp giữa các Service: REST API trên mạng nội bộ ECS

REST API (Representational State Transfer Application Programming Interface) là phương thức giao tiếp phổ biến nhờ tính đơn giản, linh hoạt và khả năng mở rộng. REST tuân theo các nguyên tắc của kiến trúc hướng tài nguyên, trong đó mỗi tài nguyên được biểu diễn bằng một định danh duy nhất (URI) và được thao tác thông qua các phương thức chuẩn của giao thức HTTP như GET, POST, PUT và DELETE. Cách tiếp cận này giúp các dịch vụ duy trì tính độc lập, giảm sự phụ thuộc lẫn nhau và tạo điều kiện thuận lợi cho việc phát triển và triển khai riêng biệt từng thành phần.

Trong hệ thống, các dịch vụ giao tiếp với nhau thông qua REST API truyền qua mạng nội bộ của ECS Cluster. Giao tiếp nội bộ mang lại nhiều lợi ích: (1) tăng cường bảo mật do không cần công khai endpoint ra Internet, (2) giảm thiểu độ trễ vì dữ liệu truyền trong hạ tầng mạng tối ưu hoá của cụm container, và (3) đảm bảo tính nhất quán trong cách các dịch vụ trao đổi dữ liệu. Việc sử dụng REST trong môi trường tách biệt cũng giúp việc giám sát, kiểm thử và mở rộng dịch vụ trở nên dễ dàng hơn, đồng thời cho phép từng dịch vụ có thể phát triển bằng công nghệ hoặc ngôn ngữ lập trình khác nhau mà không ảnh hưởng đến tổng thể hệ thống.

2.8 AWS Amplify

AWS Amplify là một nền tảng phát triển ứng dụng web và di động được xây dựng nhằm đơn giản hóa quá trình triển khai, quản lý và vận hành frontend trong môi trường đám mây. Amplify cung cấp một tập hợp các công cụ và dịch vụ giúp tự động hóa toàn bộ vòng đời phát triển của ứng dụng, từ xây dựng (build), kiểm thử (test), đến triển khai (deploy) và phân phối (hosting).

Một trong những điểm mạnh của Amplify là khả năng hỗ trợ CI/CD tích hợp, cho phép ứng dụng tự động được build và triển khai mỗi khi có thay đổi mới trên các nhánh mã nguồn. Điều này giảm thiểu sai sót thủ công, đồng thời đảm bảo rằng bản phát hành luôn nhất quán với mã nguồn mới nhất.

Ngoài ra, Amplify cung cấp hạ tầng Hosting tĩnh với CDN phân tán toàn cầu, giúp tối ưu hóa tốc độ tải trang cho người dùng ở nhiều khu vực địa lý khác nhau. Bằng việc sử dụng mạng phân phối nội dung (Content Delivery Network), các tệp frontend có thể được truy xuất nhanh chóng, giảm độ trễ và cải thiện trải nghiệm người dùng.

Về mặt bảo mật và vận hành, Amplify hỗ trợ cấu hình dễ dàng các domain tùy chỉnh, chứng chỉ HTTPS, ghi log, theo dõi hiệu năng và khả năng rollback về phiên bản trước khi gặp lỗi. Nhờ sự tích hợp chặt chẽ với các dịch vụ AWS khác, Amplify mang đến môi trường triển khai mạnh mẽ, linh hoạt và phù hợp cho các ứng dụng web hiện đại.



Hình 6: AWS Amplify

3 Yêu cầu về hệ thống

3.1 Xác định người dùng

Do hệ thống sẽ được sử dụng trong môi trường đại học, các phân loại người dùng cũng sẽ dựa trên môi trường thực tế. Điều này đảm bảo hệ thống được vận hành trơn tru khi áp dụng vào một trường đại học bất kỳ, cũng như không phải thuê thêm người điều hành riêng. Phân loại người dùng bao gồm:

- Admin: Người phụ trách vận hành, giám sát hệ thống sau khi triển khai. Nhiệm vụ của người dùng này còn bao gồm cài đặt, điều chỉnh cách vận hành của một số tính năng. Admin cũng có thể là một giảng viên trong trường.
- Trưởng khoa: Người phụ trách quản lý một số môn học của một khoa. Vai trò chủ yếu của người dùng này là kiểm duyệt các yêu cầu về quyền hạn người dùng, nội dung của các khóa học.
- Giảng viên: Là người phụ trách đăng tải tài liệu, xây dựng bài học, quiz,... Giảng viên cũng là người sẽ trả lời các câu hỏi trên diễn đàn của sinh viên, theo dõi tiến độ học tập,...
- Trợ giảng (TA): Vai trò chủ yếu của người dùng này là trợ giúp các giảng viên về các vấn đề liên quan tới một khóa học, như đề xuất chỉnh sửa nội dung, thảo luận trên diễn đàn, theo dõi tiến độ sinh viên,...
- Sinh viên: Là người dùng trực tiếp sử dụng các tài nguyên được đăng tải trên hệ thống. Sinh viên có quyền được truy cập các khóa học, xem nội dung khóa học, làm bài tập,...

3.2 Yêu cầu chức năng

Biểu diễn các yêu cầu chức năng dưới dạng User story, ta có bảng sau:

Mã	Ai	Muốn gì	Để làm gì
A-US01	Admin	Quản lý người dùng (tạo, xóa, sửa)	Đảm bảo hệ thống vận hành ổn định và đúng đối tượng sử dụng
A-US02	Admin	Quản lý quyền (thêm, xóa)	Đảm bảo hệ thống vận hành ổn định và đúng đối tượng sử dụng
A-US03	Admin	Cấu hình các tham số hệ thống	Tinh chỉnh hoạt động hệ thống theo yêu cầu thực tế
A-US04	Admin	Theo dõi hoạt động hệ thống (logs, thống kê)	Giám sát, phát hiện lỗi và tối ưu hiệu năng
D-US01	Trưởng khoa	Xác thực yêu cầu cấp quyền cho giảng viên hoặc trợ giảng	Đảm bảo đúng người đúng vai trò trong khoa
D-US02	Trưởng khoa	Tạo, kiểm duyệt nội dung các khóa học của khoa	Đảm bảo chất lượng và tính phù hợp chuyên môn
L-US01	Giảng viên	Yêu cầu quyền truy cập khóa học	Giới hạn chỉnh sửa các khóa học được cấp phép
L-US02	Giảng viên	Tạo và quản lý nội dung khóa học (bài giảng, bài tập, quiz)	Xây dựng tài liệu giúp sinh viên học tập hiệu quả
L-US03	Giảng viên	Theo dõi tiến độ học tập của sinh viên	Đánh giá và hỗ trợ học tập kịp thời
L-US04	Giảng viên	Trả lời câu hỏi trên diễn đàn khóa học	Giải đáp thắc mắc cho sinh viên
T-US01	Trợ giảng	Yêu cầu quyền truy cập khóa học	Giới hạn chỉnh sửa các khóa học được cấp phép
T-US02	Trợ giảng	Hỗ trợ giảng viên chỉnh sửa hoặc đề xuất nội dung khóa học	Giảm tải công việc và nâng cao chất lượng khóa học
T-US03	Trợ giảng	Tham gia thảo luận trên diễn đàn	Hỗ trợ giải đáp câu hỏi của sinh viên
T-US04	Trợ giảng	Theo dõi tiến độ sinh viên	Hỗ trợ giảng viên trong đánh giá và nhắc nhở sinh viên
S-US01	Sinh viên	Truy cập và xem nội dung khóa học	Học tập theo chương trình giảng viên cung cấp
S-US02	Sinh viên	Làm bài tập, quiz và xem kết quả	Đánh giá mức độ hiểu bài và cải thiện kiến thức
S-US03	Sinh viên	Đặt câu hỏi và thảo luận trên diễn đàn	Trao đổi với giảng viên, trợ giảng và bạn học
S-US04	Sinh viên	Theo dõi tiến độ học tập của bản thân	Quản lý quá trình học tập và hoàn thành yêu cầu khóa học

Bảng 1: Bảng User Stories

3.3 Yêu cầu phi chức năng

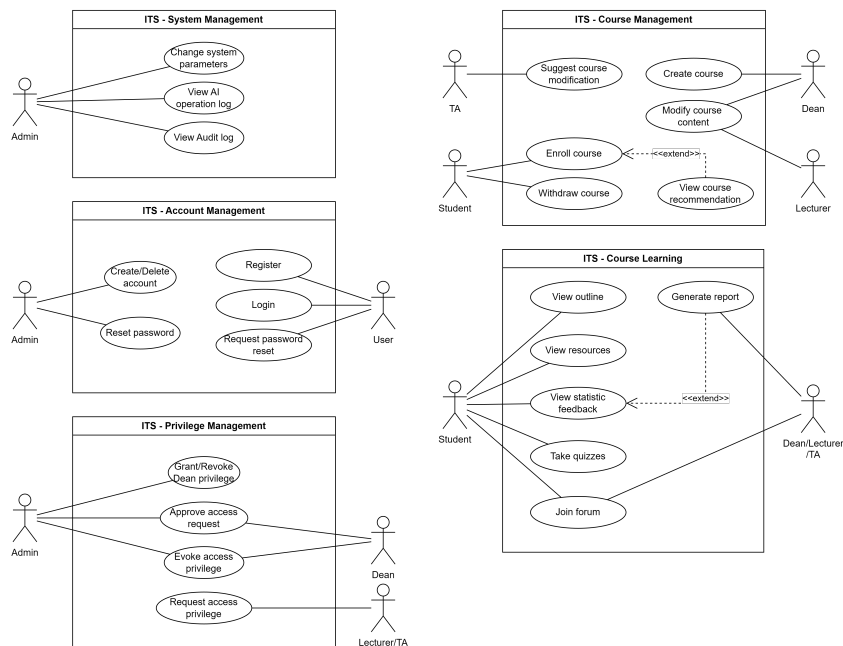
Từ kết quả thăm dò ý kiến người dùng, kết hợp brainstorm trong các buổi họp nhóm, các yêu cầu phi chức năng của hệ thống được miêu tả như sau:

Mã NFR	Mô tả yêu cầu phi chức năng
NRF01	Hệ thống phải cho phép thay đổi hành vi vận hành mà không cần chỉnh sửa mã nguồn, bao gồm bật/tắt tính năng và điều chỉnh tham số hoạt động.
NRF02	Hệ thống phải hỗ trợ thay đổi mô hình AI mà không gây gián đoạn dịch vụ (zero-downtime model update).
NRF03	Hệ thống phải cho phép cấu hình loại tài liệu được phép đăng tải trong khóa học mà không cần triển khai lại.
NRF04	Hệ thống phải cho phép chỉnh sửa layout hoặc cấu trúc giao diện khóa học thông qua cơ chế cấu hình.
NRF05	Module tạo báo cáo phải hỗ trợ cấu hình layout của báo cáo.
NRF06	Module tạo báo cáo phải cho phép cấu hình phạm vi dữ liệu được sử dụng trong báo cáo.
NRF07	Module tạo quiz bằng AI phải hỗ trợ cấu hình số lượng câu hỏi, độ khó và phân bố độ khó.
NRF08	Module gợi ý hint phải cho phép cấu hình tần suất và mức độ chi tiết của gợi ý.
NRF09	Module gợi ý chủ đề phải cho phép thêm hoặc loại bỏ chủ đề quan tâm thông qua cấu hình.
NRF10	Hệ thống phải đảm bảo mức độ sẵn sàng không dưới 99% trong thời gian có lưu lượng truy cập cao.
NRF11	Khi xảy ra sự cố downtime, hệ thống phải tự phục hồi trong thời gian không vượt quá 5 phút.
NRF12	Các thành phần quan trọng phải được triển khai theo mô hình nhiều vùng sẵn sàng (multi-AZ) hoặc tương đương.
NRF13	Dữ liệu bài giảng đăng tải phải được kiểm tra tính hợp lệ (định dạng, nội dung đầy đủ).
NRF14	Mọi dữ liệu tải lên phải được lưu trữ an toàn, bao gồm mã hóa khi lưu trữ và khi truyền tải.
NRF15	Mọi dữ liệu hiển thị phải đảm bảo nhất quán giữa các người dùng và các phiên xử lý.
NRF16	Các thao tác quan trọng phải đảm bảo tính toàn vẹn giao dịch (transactional integrity).
NRF17	Lịch sử chỉnh sửa nội dung khóa học phải được ghi lại để phục vụ truy vết.
NRF18	Các API liên quan đến AI phải có thời gian phản hồi trung bình dưới 2 giây.
NRF19	Thời gian tải nội dung khóa học không vượt quá 1 giây trên kết nối mạng phổ thông.
NRF20	Hệ thống phải sử dụng cơ chế caching để cải thiện tốc độ truy xuất dữ liệu.
NRF21	Hệ thống phải hỗ trợ ít nhất X sinh viên truy cập đồng thời (giá trị X được xác định sau khi ước tính tải).
NRF22	Hệ thống phải tránh xung đột dữ liệu (race condition) khi nhiều người dùng thao tác song song.
NRF23	Hệ thống phải tự động mở rộng tài nguyên khi tải tăng cao, đặc biệt trong mùa thi.
NRF24	Hệ thống phải tự động thu hẹp tài nguyên khi tải giảm để tối ưu chi phí vận hành.
NRF25	Các dịch vụ AI và quiz phải hỗ trợ autoscaling dựa trên CPU/GPU hoặc số lượng yêu cầu mỗi giây.

Bảng 2: Danh sách yêu cầu phi chức năng (NFR)

4 Đặc tả chi tiết các use-case

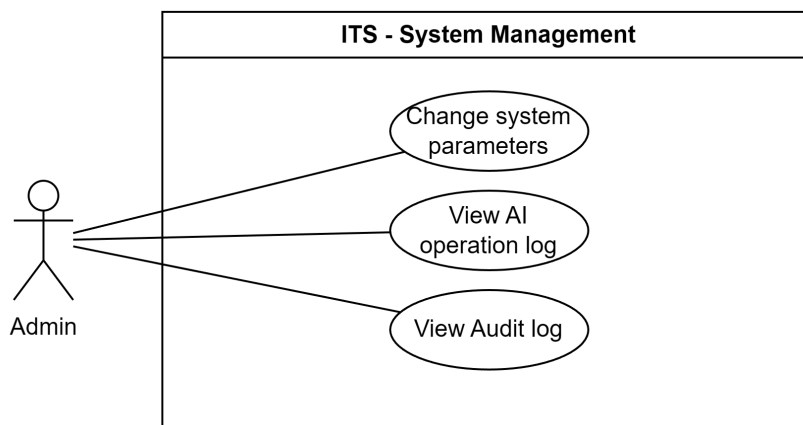
Để thiết kế kiến trúc của hệ thống, nhóm đã hợp và triển khai các User story thành các use-case cụ thể.



Hình 7: Tổng quan các Use-case

4.1 Quản lý hệ thống

4.1.1 Tổng quan



Hình 8: Các use-case về Quản lý hệ thống

4.1.2 Đặc tả chi tiết

Bảng 3: Usecase «Cấu hình hệ thống»

Usecase	Cấu hình hệ thống
Nguồn	A-US03, NFR01
Use-case ID	UC-SM-01
Tổng quan	Admin chỉnh sửa các tham số hệ thống để tinh chỉnh hoạt động
Actor	Admin

Usecase	Cấu hình hệ thống
Tiền điều kiện	<ul style="list-style-type: none">Hệ thống đang hoạt động ổn định.Admin đã đăng nhập và có quyền.
Điều kiện kích hoạt	<ul style="list-style-type: none">Admin truy cập vào trang cấu hình hệ thống.
Các bước	<ol style="list-style-type: none">Hiển thị giao diện cấu hình tham số hệ thống.Admin thay đổi các tham số cấu hình (lập lịch trích xuất, quản lý phạm vi trích xuất, mức hoạt động tối thiểu, hệ số đa dạng đề xuất).Admin lưu thay đổi và xác nhận.Hệ thống cập nhật và áp dụng cấu hình mới.
Hậu điều kiện	<ul style="list-style-type: none">Các tham số hệ thống được cập nhật thành công.Dịch vụ liên quan áp dụng tham số mới.
Xử lý ngoại lệ	ex1. Không có thay đổi khi lưu, hệ thống hỏi xác nhận. ex2. Lỗi lưu trữ, hiển thị thông báo lỗi.

Bảng 4: Usecase «Xem nhật ký hoạt động AI»

Usecase	Xem nhật ký hoạt động AI
Nguồn	A-US04, NFR18
Use-case ID	UC-SM-02
Tổng quan	Admin xem nhật ký hoạt động (logs) của hệ thống AI để giám sát và phát hiện sự cố.
Actor	Admin
Tiền điều kiện	<ul style="list-style-type: none">Admin đã đăng nhập với quyền xem nhật ký.
Điều kiện kích hoạt	<ul style="list-style-type: none">Admin truy cập trang nhật ký hoạt động AI.
Các bước	<ol style="list-style-type: none">Hệ thống hiển thị danh sách các bản ghi nhật ký AI, hỗ trợ lọc theo thời gian và mức độ nghiêm trọng. (Message, Warning, Error)Admin lựa chọn bản ghi hoặc khoảng thời gian cần xem.Hệ thống hiển thị chi tiết nhật ký theo yêu cầu.
Hậu điều kiện	<ul style="list-style-type: none">Admin có thể nắm bắt được thông tin hoạt động, cảnh báo hoặc lỗi của hệ thống AI.
Xử lý ngoại lệ	ex1. Nếu không có dữ liệu nhật ký trong khoảng thời gian yêu cầu, hệ thống hiển thị thông báo "Không có dữ liệu". ex2. Nếu xảy ra lỗi truy xuất dữ liệu nhật ký (ví dụ: mất kết nối), hệ thống thông báo lỗi và gợi ý tải lại trang.

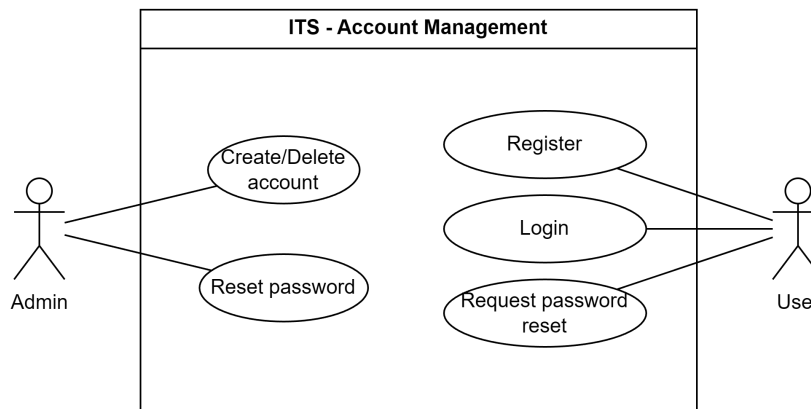
Bảng 5: Usecase «Xem nhật ký kiểm toán (Audit log)»

Usecase	Xem nhật ký kiểm toán (Audit log)
Nguồn	A-US04
Use-case ID	UC-SM-03
Tổng quan	Admin xem nhật ký kiểm toán để giám sát các hành động hệ thống, đảm bảo an toàn và phát hiện hành vi bất thường.

Usecase	Xem nhật ký kiểm toán (Audit log)
Actor	Admin
Tiền điều kiện	<ul style="list-style-type: none"> Admin đã đăng nhập với quyền xem audit log.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Admin truy cập trang nhật ký kiểm toán.
Các bước	<ol style="list-style-type: none"> Hệ thống hiển thị danh sách các sự kiện audit log, hỗ trợ lọc theo người dùng, loại sự kiện, thời gian. Admin chọn bộ lọc hoặc tìm kiếm cụ thể. Hệ thống hiển thị chi tiết các sự kiện phù hợp.
Hậu điều kiện	<ul style="list-style-type: none"> Admin có thể kiểm tra và truy vết các thao tác hệ thống quan trọng.
Xử lý ngoại lệ	<p>ex1. Nếu không có sự kiện nào phù hợp bộ lọc, hệ thống hiển thị thông báo "Không tìm thấy sự kiện".</p> <p>ex2. Nếu lỗi truy xuất dữ liệu audit log, hệ thống thông báo lỗi và yêu cầu tải lại.</p>

4.2 Quản lý tài khoản

4.2.1 Tổng quan



Hình 9: Các use-case về Quản lý tài khoản

4.2.2 Đặc tả chi tiết

Bảng 6: Usecase «Tạo hoặc Xóa tài khoản»

Usecase	Tạo hoặc Xóa tài khoản
Nguồn	A-US01
Use-case ID	UC-AM-01
Tổng quan	Admin quản lý tài khoản người dùng: tạo mới hoặc xóa tài khoản khi cần thiết.
Actor	Admin
Tiền điều kiện	<ul style="list-style-type: none"> Admin đã đăng nhập với quyền quản lý tài khoản.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Admin truy cập vào trang quản lý tài khoản.

Usecase	Tạo hoặc Xóa tài khoản
Các bước	<ol style="list-style-type: none">Admin chọn thao tác tạo tài khoản mới hoặc xóa tài khoản hiện có.Nếu tạo, Admin nhập các thông tin cần thiết (tên, email, quyền hạn,...).Hệ thống kiểm tra dữ liệu hợp lệ và tạo tài khoản.Nếu xóa, Admin chọn tài khoản cần xóa và xác nhận.Hệ thống xóa tài khoản và thông báo thành công.
Hậu điều kiện	<ul style="list-style-type: none">Tài khoản được tạo mới hoặc xóa khỏi hệ thống thành công.
Xử lý ngoại lệ	<p>ex1. Nếu dữ liệu nhập không hợp lệ (ví dụ: email trùng, thiếu thông tin bắt buộc), hệ thống thông báo lỗi.</p> <p>ex2. Nếu tài khoản cần xóa đang liên kết với dữ liệu quan trọng (ví dụ: đang quản lý khóa học), hệ thống cảnh báo và yêu cầu xác nhận thêm hoặc từ chối xóa.</p> <p>ex3. Nếu lỗi hệ thống khi lưu hoặc xóa, thông báo lỗi và gợi ý thử lại.</p>

Bảng 7: Usecase «Đặt lại mật khẩu (Reset password)»

Usecase	Đặt lại mật khẩu (Reset password)
Nguồn	A-US01
Use-case ID	UC-AM-02
Tổng quan	Admin có thể đặt lại mật khẩu cho tài khoản người dùng khi cần thiết.
Actor	Admin
Tiền điều kiện	<ul style="list-style-type: none">Admin đã đăng nhập.Admin có quyền đặt lại mật khẩu.
Điều kiện kích hoạt	<ul style="list-style-type: none">Admin truy cập trang quản lý tài khoản và chọn đặt lại mật khẩu.
Các bước	<ol style="list-style-type: none">Admin chọn tài khoản người dùng cần đặt lại mật khẩu.Nhập mật khẩu mới hoặc hệ thống sinh mật khẩu ngẫu nhiên.Xác nhận thao tác đặt lại mật khẩu.Hệ thống cập nhật mật khẩu và thông báo thành công.
Hậu điều kiện	<ul style="list-style-type: none">Mật khẩu của tài khoản được thay đổi và có hiệu lực ngay lập tức.
Xử lý ngoại lệ	<p>ex1. Nếu tài khoản không tồn tại hoặc không hợp lệ, hệ thống thông báo lỗi.</p> <p>ex2. Nếu lỗi hệ thống khi cập nhật, thông báo lỗi và yêu cầu thử lại.</p>

Bảng 8: Usecase «Đăng ký tài khoản»

Usecase	Đăng ký tài khoản
Nguồn	
Use-case ID	UC-AM-03
Tổng quan	Người dùng mới có thể đăng ký tài khoản để sử dụng hệ thống.
Actor	User
Tiền điều kiện	<ul style="list-style-type: none">Người dùng chưa có tài khoản trong hệ thống.

Usecase	Đăng ký tài khoản
Điều kiện kích hoạt	<ul style="list-style-type: none">• Người dùng truy cập trang đăng ký tài khoản.
Các bước	<ol style="list-style-type: none">1. Người dùng nhập thông tin cá nhân và tài khoản (email, mật khẩu, tên,...).2. Hệ thống kiểm tra tính hợp lệ của dữ liệu.3. Hệ thống tạo tài khoản mới và gửi email xác nhận (nếu có).4. Người dùng kích hoạt tài khoản qua email (nếu áp dụng).
Hậu điều kiện	<ul style="list-style-type: none">• Người dùng có thể đăng nhập và sử dụng hệ thống.
Xử lý ngoại lệ	<p>ex1. Nếu email đã được đăng ký, hệ thống báo lỗi.</p> <p>ex2. Nếu dữ liệu không hợp lệ, yêu cầu nhập lại.</p> <p>ex3. Nếu lỗi gửi email xác nhận, cảnh báo và hướng dẫn liên hệ quản trị.</p>

Bảng 9: Usecase «Đăng nhập»

Usecase	Đăng nhập
Nguồn	A-US01
Use-case ID	UC-AM-04
Tổng quan	Người dùng đăng nhập vào hệ thống để sử dụng các chức năng.
Actor	User
Tiền điều kiện	<ul style="list-style-type: none">• Người dùng đã có tài khoản hợp lệ.
Điều kiện kích hoạt	<ul style="list-style-type: none">• Người dùng truy cập trang đăng nhập.
Các bước	<ol style="list-style-type: none">1. Người dùng nhập thông tin đăng nhập (tên đăng nhập và mật khẩu).2. Hệ thống xác thực thông tin.3. Nếu thành công, hệ thống cho phép truy cập vào hệ thống.4. Nếu thất bại, hiển thị thông báo lỗi và hướng dẫn thử lại.
Hậu điều kiện	<ul style="list-style-type: none">• Người dùng có phiên làm việc hợp lệ.
Xử lý ngoại lệ	<p>ex1. Nếu tài khoản bị khóa hoặc chưa kích hoạt, hệ thống thông báo trạng thái.</p> <p>ex2. Nếu lỗi hệ thống khi xác thực, thông báo lỗi và yêu cầu thử lại.</p>

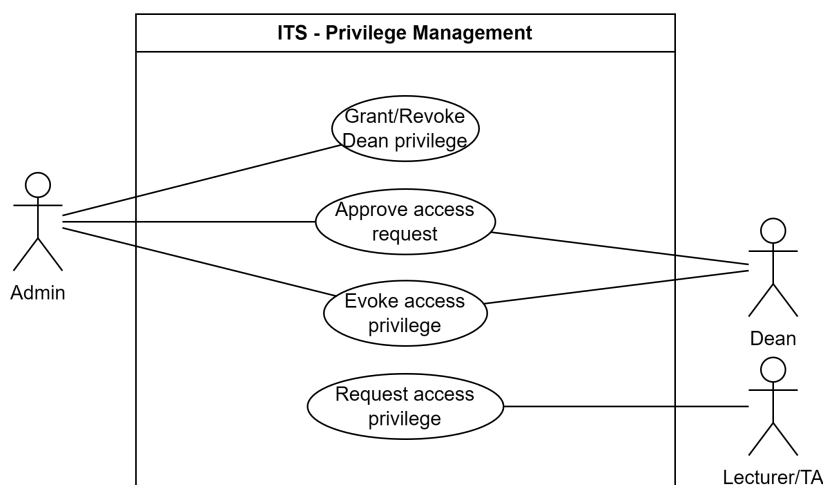
Bảng 10: Usecase «Yêu cầu đặt lại mật khẩu»

Usecase	Yêu cầu đặt lại mật khẩu
Nguồn	
Use-case ID	UC-AM-05
Tổng quan	Người dùng yêu cầu hệ thống gửi hướng dẫn đặt lại mật khẩu khi quên mật khẩu.
Actor	User
Tiền điều kiện	<ul style="list-style-type: none">• Người dùng có tài khoản đã đăng ký.
Điều kiện kích hoạt	<ul style="list-style-type: none">• Người dùng truy cập trang "Quên mật khẩu" và nhập email đăng ký.

Usecase	Yêu cầu đặt lại mật khẩu
Các bước	<ol style="list-style-type: none"> 1. Người dùng nhập địa chỉ email đã đăng ký. 2. Hệ thống kiểm tra email có tồn tại trong hệ thống. 3. Nếu có, hệ thống gửi tạo yêu cầu đặt lại mật khẩu cho Admin 4. Hệ thống hiển thị thông báo gửi yêu cầu thành công.
Hậu điều kiện	<ul style="list-style-type: none"> • Yêu cầu đặt lại mật khẩu được tạo.
Xử lý ngoại lệ	ex1. Nếu email không tồn tại, hệ thống thông báo không tìm thấy tài khoản.

4.3 Quản lý quyền truy cập

4.3.1 Tổng quan



Hình 10: Các use-case về Quản lý quyền chỉnh sửa khóa học

4.3.2 Đặc tả chi tiết

Bảng 11: Usecase «Cấp và thu hồi quyền Trưởng khoa»

Usecase	Cấp và thu hồi quyền Trưởng khoa
Nguồn	A-US02
Use-case ID	UC-PM-01
Tổng quan	Admin thực hiện cấp hoặc thu hồi quyền Trưởng khoa cho người dùng phù hợp để đảm bảo phân quyền đúng đối tượng.
Actor	Admin
Tiền điều kiện	<ul style="list-style-type: none"> • Admin đã đăng nhập. • Admin có quyền quản lý quyền truy cập. • Người dùng cần cấp/thu hồi quyền tồn tại trong hệ thống.
Điều kiện kích hoạt	<ul style="list-style-type: none"> • Admin truy cập trang quản lý quyền và chọn chức năng cấp hoặc thu hồi quyền Trưởng khoa.

Usecase	Cấp và thu hồi quyền Trưởng khoa
Các bước	<ol style="list-style-type: none"> Admin tìm kiếm hoặc chọn người dùng cần thay đổi quyền. Admin chọn cấp quyền Trưởng khoa hoặc thu hồi quyền. Xác nhận thao tác cấp hoặc thu hồi. Hệ thống cập nhật quyền truy cập cho người dùng. Hệ thống hiển thị thông báo thực hiện thành công.
Hậu điều kiện	<ul style="list-style-type: none"> Quyền Trưởng khoa được cập nhật chính xác theo yêu cầu.
Xử lý ngoại lệ	<p>ex1. Nếu người dùng không tồn tại hoặc đã có quyền tương ứng, hệ thống thông báo lỗi và ngăn thao tác.</p> <p>ex2. Nếu lỗi hệ thống khi cập nhật quyền, hệ thống hiển thị lỗi và yêu cầu thử lại.</p> <p>ex3. Nếu Admin không có quyền thực hiện thao tác, hệ thống từ chối truy cập.</p>

Bảng 12: Usecase «Phê duyệt yêu cầu quyền truy cập»

Usecase	Phê duyệt yêu cầu quyền truy cập
Nguồn	A-US02, D-US01
Use-case ID	UC-PM-02
Tổng quan	Admin hoặc Trưởng khoa xem và phê duyệt các yêu cầu cấp quyền truy cập của giảng viên hoặc trợ giảng.
Actor	Admin / Dean (Trưởng khoa)
Tiền điều kiện	<ul style="list-style-type: none"> Admin hoặc Trưởng khoa đã đăng nhập với quyền phê duyệt. Có các yêu cầu quyền truy cập đang chờ xử lý.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Người có quyền truy cập vào trang phê duyệt yêu cầu quyền.
Các bước	<ol style="list-style-type: none"> Xem danh sách các yêu cầu cấp quyền đang chờ. Chọn từng yêu cầu để xem chi tiết. Phê duyệt hoặc từ chối yêu cầu. Hệ thống cập nhật trạng thái và thông báo kết quả cho người yêu cầu.
Hậu điều kiện	<ul style="list-style-type: none"> Quyền truy cập được cấp hoặc yêu cầu bị từ chối chính xác. Người yêu cầu được thông báo kết quả kịp thời.
Xử lý ngoại lệ	<p>ex1. Nếu yêu cầu không tồn tại hoặc đã được xử lý, hệ thống thông báo và không thực hiện thao tác.</p> <p>ex2. Nếu lỗi cập nhật quyền truy cập, hệ thống báo lỗi và cho phép thử lại.</p> <p>ex3. Nếu người phê duyệt không có quyền hợp lệ, hệ thống từ chối thao tác.</p>

Bảng 13: Usecase «Thu hồi quyền truy cập»

Usecase	Thu hồi quyền truy cập
Nguồn	A-US02, D-US01
Use-case ID	UC-PM-03
Tổng quan	Admin hoặc Trưởng khoa thu hồi quyền truy cập của giảng viên hoặc trợ giảng khi không còn phù hợp.
Actor	Admin / Dean (Trưởng khoa)

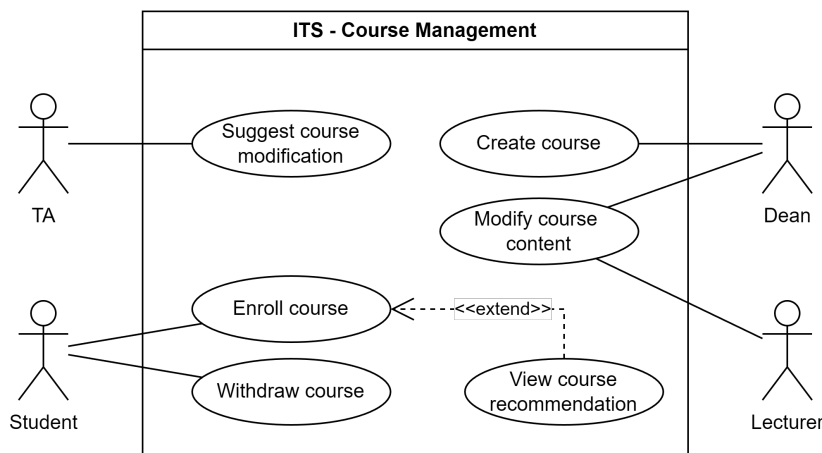
Usecase	Thu hồi quyền truy cập
Tiền điều kiện	<ul style="list-style-type: none"> Người thực hiện đã đăng nhập với quyền thu hồi quyền. Người dùng bị thu hồi quyền đang có quyền hợp lệ trong hệ thống.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Người thực hiện truy cập trang quản lý quyền và chọn thu hồi quyền.
Các bước	<ol style="list-style-type: none"> Tìm hoặc chọn người dùng cần thu hồi quyền. Chọn quyền cần thu hồi. Xác nhận thao tác thu hồi. Hệ thống cập nhật và ghi nhận thay đổi. Hiển thị thông báo thu hồi quyền thành công.
Hậu điều kiện	<ul style="list-style-type: none"> Quyền truy cập bị thu hồi có hiệu lực ngay lập tức.
Xử lý ngoại lệ	<p>ex1. Nếu người dùng không tồn tại hoặc không có quyền được thu hồi, thông báo lỗi.</p> <p>ex2. Nếu lỗi hệ thống khi cập nhật, hiển thị thông báo và đề nghị thử lại.</p> <p>ex3. Nếu người thực hiện không đủ quyền, hệ thống từ chối thao tác.</p>

Bảng 14: Usecase «Yêu cầu quyền truy cập»

Usecase	Yêu cầu quyền truy cập
Nguồn	L-US01, T-US01
Use-case ID	UC-PM-04
Tổng quan	Giảng viên hoặc trợ giảng gửi yêu cầu xin cấp quyền truy cập để thực hiện các chức năng nâng cao.
Actor	Lecturer / TA (Giảng viên / Trợ giảng)
Tiền điều kiện	<ul style="list-style-type: none"> Người dùng đã đăng nhập và chưa có quyền cần xin cấp.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Người dùng truy cập trang yêu cầu quyền truy cập.
Các bước	<ol style="list-style-type: none"> Người dùng chọn loại quyền cần xin cấp. Nhập lý do hoặc mô tả nhu cầu sử dụng quyền. Gửi yêu cầu lên hệ thống. Hệ thống lưu yêu cầu và thông báo đã gửi thành công.
Hậu điều kiện	<ul style="list-style-type: none"> Yêu cầu quyền được ghi nhận và chờ phê duyệt.
Xử lý ngoại lệ	<p>ex1. Nếu người dùng đã có quyền tương ứng, hệ thống báo không cần gửi yêu cầu.</p> <p>ex2. Nếu dữ liệu gửi lên không hợp lệ (ví dụ: thiếu lý do), hệ thống thông báo lỗi.</p> <p>ex3. Nếu lỗi hệ thống khi lưu yêu cầu, hệ thống báo lỗi và đề nghị thử lại.</p>

4.4 Quản lý khóa học

4.4.1 Tổng quan



Hình 11: Các use-case về Quản lý khóa học

4.4.2 Đặc tả chi tiết

Bảng 15: Usecase «Đề xuất chỉnh sửa khóa học»

Usecase	Đề xuất chỉnh sửa khóa học
Nguồn	T-US02
Use-case ID	UC-CM-01
Tổng quan	Trợ giảng đề xuất chỉnh sửa nội dung khóa học để nâng cao chất lượng.
Actor	TA (Trợ giảng)
Tiền điều kiện	<ul style="list-style-type: none"> Trợ giảng đã đăng nhập và có quyền truy cập vào khóa học được phân công.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Trợ giảng truy cập trang khóa học và chọn chức năng đề xuất chỉnh sửa.
Các bước	<ol style="list-style-type: none"> Trợ giảng xem nội dung khóa học hiện tại. Trợ giảng chỉnh sửa hoặc nhập đề xuất thay đổi. Gửi đề xuất cho trưởng khoa hoặc giảng viên xem xét. Hệ thống lưu và thông báo trạng thái đề xuất.
Hậu điều kiện	<ul style="list-style-type: none"> Đề xuất được ghi nhận và chờ phê duyệt.
Xử lý ngoại lệ	<p>ex1. Nếu dữ liệu đề xuất không hợp lệ, hệ thống thông báo lỗi.</p> <p>ex2. Nếu lỗi lưu trữ, thông báo lỗi và yêu cầu thử lại.</p>

Bảng 16: Usecase «Tạo khóa học»

Usecase	Tạo khóa học
Nguồn	D-US02
Use-case ID	UC-CM-02
Tổng quan	Trưởng khoa tạo mới khóa học thuộc khoa mình quản lý.

Usecase	Tạo khóa học
Actor	Dean (Trưởng khoa)
Tiền điều kiện	<ul style="list-style-type: none"> Trưởng khoa đã đăng nhập và có quyền tạo khóa học.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Trưởng khoa truy cập trang quản lý khóa học và chọn tạo mới.
Các bước	<ol style="list-style-type: none"> Nhập các thông tin khóa học (tên, mã, mô tả,...). Xác nhận thông tin và lưu khóa học mới. Hệ thống tạo khóa học và hiển thị thông báo thành công.
Hậu điều kiện	<ul style="list-style-type: none"> Khóa học được thêm mới trong hệ thống và sẵn sàng cho giảng viên quản lý.
Xử lý ngoại lệ	<p>ex1. Nếu thông tin không hợp lệ hoặc trùng mã khóa học, thông báo lỗi.</p> <p>ex2. Nếu lỗi hệ thống khi lưu, thông báo lỗi và yêu cầu thử lại.</p>

Bảng 17: Usecase «Sửa nội dung khóa học»

Usecase	Sửa nội dung khóa học
Nguồn	L-US01
Use-case ID	UC-CM-03
Tổng quan	Giảng viên sửa đổi nội dung khóa học để cập nhật bài giảng, bài tập, quiz,...
Actor	Lecturer (Giảng viên)
Tiền điều kiện	<ul style="list-style-type: none"> Giảng viên đã đăng nhập và được phân quyền chỉnh sửa khóa học.
Điều kiện kích hoạt	<ul style="list-style-type: none"> Giảng viên truy cập trang quản lý khóa học và chọn chỉnh sửa.
Các bước	<ol style="list-style-type: none"> Hiển thị nội dung khóa học hiện tại (các chương, phần trong chương, chủ đề của phần, tài li) Giảng viên thực hiện chỉnh sửa các phần nội dung. Lưu thay đổi và hệ thống cập nhật nội dung. Hệ thống ghi lại lịch sử chỉnh sửa để truy vết.
Hậu điều kiện	<ul style="list-style-type: none"> Nội dung khóa học được cập nhật và hiển thị cho sinh viên.
Xử lý ngoại lệ	<p>ex1. Nếu dữ liệu nhập không hợp lệ, cảnh báo và yêu cầu chỉnh sửa.</p> <p>ex2. Nếu lỗi lưu dữ liệu, hệ thống thông báo và cho phép thử lại.</p>

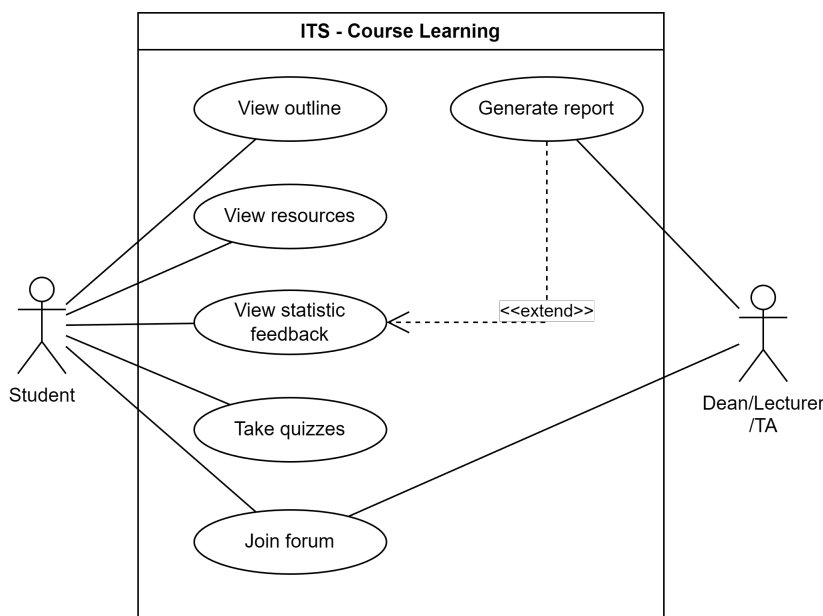
Bảng 18: Usecase «Ghi danh khóa học»

Usecase	Ghi danh khóa học
Nguồn	S-US01
Use-case ID	UC-CM-04
Tổng quan	Sinh viên ghi danh tham gia khóa học.
Actor	Student (Sinh viên)
Tiền điều kiện	<ul style="list-style-type: none"> Sinh viên đã đăng nhập. Khóa học đang mở ghi danh.

Usecase	Ghi danh khóa học
Điều kiện kích hoạt	<ul style="list-style-type: none"> Sinh viên chọn khóa học và chọn ghi danh.
Các bước	<ol style="list-style-type: none"> Hiển thị thông tin khóa học. Sinh viên nhấn ghi danh. Hệ thống kiểm tra điều kiện và ghi nhận đăng ký. Hiển thị xác nhận ghi danh thành công.
Hậu điều kiện	<ul style="list-style-type: none"> Sinh viên được ghi danh và có thể truy cập nội dung khóa học.
Xử lý ngoại lệ	<p>ex1. Nếu khóa học không cho phép ghi danh, thông báo lỗi.</p> <p>ex2. Nếu sinh viên đã ghi danh trước đó, thông báo không thể ghi danh lại.</p> <p>ex3. Nếu lỗi hệ thống khi ghi danh, cảnh báo và đề nghị thử</p>

4.5 Học khóa học

4.5.1 Tổng quan



Hình 12: Các use-case về Học khóa học

4.5.2 Đặc tả chi tiết

Bảng 19: Usecase «Xem đề cương khóa học»

Usecase	Xem đề cương khóa học
Nguồn	S-US01, NFR19
Use-case ID	UC-CL-01
Tổng quan	Sinh viên xem đề cương chi tiết của khóa học để nắm được nội dung và kế hoạch học tập.
Actor	Student
Tiền điều kiện	<ul style="list-style-type: none"> Sinh viên đã đăng nhập và đã ghi danh khóa học. Đề cương khóa học đã được cập nhật trên hệ thống.

Usecase	Xem đề cương khóa học
Điều kiện kích hoạt	Sinh viên chọn khóa học và yêu cầu xem đề cương.
Các bước	1. Hệ thống hiển thị danh sách khóa học mà sinh viên đã đăng ký. 2. Sinh viên chọn khóa học muốn xem đề cương. 3. Hệ thống tải và hiển thị đề cương khóa học.
Hậu điều kiện	• Sinh viên nhận được đề cương đầy đủ, rõ ràng.
Xử lý ngoại lệ	ex1. Sinh viên chưa đăng ký khóa học, hệ thống thông báo không có quyền truy cập. ex2. Đề cương chưa được cập nhật hoặc lỗi tải, hệ thống thông báo lỗi và hướng dẫn liên hệ quản trị. ex3. Lỗi kết nối mạng hoặc hệ thống, hệ thống yêu cầu tải lại trang hoặc thử lại sau.

Bảng 20: Usecase «Xem tài liệu học tập»

Usecase	Xem tài liệu học tập
Nguồn	S-US01, NFR19, NFR20
Use-case ID	UC-CL-02
Tổng quan	Sinh viên truy cập và sử dụng tài liệu học tập được cung cấp trong khóa học.
Actor	Student
Tiền điều kiện	• Sinh viên đã đăng nhập và ghi danh khóa học. • Tài liệu học tập đã được giảng viên hoặc trợ giảng đăng tải hợp lệ.
Điều kiện kích hoạt	Sinh viên chọn mục tài liệu học tập trong khóa học.
Các bước	1. Hệ thống liệt kê tài liệu học tập hiện có trong khóa học. 2. Sinh viên chọn tài liệu cần xem hoặc tải về. 3. Hệ thống hiển thị hoặc cung cấp file tải cho sinh viên.
Hậu điều kiện	• Sinh viên tiếp cận tài liệu học tập đầy đủ, thuận tiện.
Xử lý ngoại lệ	ex1. Tài liệu bị thiếu hoặc lỗi tải, hệ thống thông báo và đề nghị liên hệ quản trị. ex2. Sinh viên không có quyền truy cập do chưa đăng ký khóa học, hệ thống thông báo rõ. ex3. Lỗi mạng hoặc lỗi hệ thống, hệ thống hỗ trợ tải lại hoặc báo lỗi tạm thời.

Bảng 21: Usecase «Xem phản hồi thống kê»

Usecase	Xem phản hồi thống kê
Nguồn	S-US04, NFR19
Use-case ID	UC-CL-03
Tổng quan	Sinh viên xem các thống kê về tiến độ học tập và kết quả để theo dõi sự tiến bộ của bản thân.
Actor	Student

Usecase	Xem phản hồi thống kê
Tiền điều kiện	<ul style="list-style-type: none"> Sinh viên đã ghi danh và có dữ liệu học tập trên hệ thống. Hệ thống đã tổng hợp dữ liệu thống kê tiến độ học tập.
Điều kiện kích hoạt	Sinh viên truy cập trang thống kê học tập.
Các bước	<ol style="list-style-type: none"> Hệ thống hiển thị các số liệu thống kê (điểm quiz, bài tập, tiến độ khóa học). Sinh viên xem và phân tích kết quả.
Hậu điều kiện	<ul style="list-style-type: none"> Sinh viên có thông tin đầy đủ để điều chỉnh kế hoạch học tập.
Xử lý ngoại lệ	<p>ex1. Dữ liệu thống kê chưa đủ hoặc bị lỗi, hệ thống thông báo.</p> <p>ex2. Lỗi hệ thống khi tải dữ liệu, hiển thị thông báo lỗi.</p>

Bảng 22: Usecase «Làm bài kiểm tra (quiz)»

Usecase	Làm bài kiểm tra (quiz)
Nguồn	S-US02, NFR07, NFR18
Use-case ID	UC-CL-04
Tổng quan	Sinh viên làm các bài kiểm tra để đánh giá kiến thức đã học.
Actor	Student
Tiền điều kiện	<ul style="list-style-type: none"> Sinh viên đã đăng ký khóa học và được phép làm quiz. Quiz đã được tạo và kích hoạt.
Điều kiện kích hoạt	Sinh viên chọn làm bài kiểm tra trong khóa học.
Các bước	<ol style="list-style-type: none"> Hệ thống hiển thị danh sách bài quiz có thể làm. Sinh viên bắt đầu làm bài, trả lời các câu hỏi. Hệ thống ghi nhận và đánh giá câu trả lời. Kết thúc bài kiểm tra, hệ thống hiển thị kết quả và phản hồi.
Hậu điều kiện	<ul style="list-style-type: none"> Điểm số và kết quả được lưu trữ để theo dõi và đánh giá.
Xử lý ngoại lệ	<p>ex1. Quiz hết hạn hoặc không tồn tại, hệ thống thông báo không thể làm bài.</p> <p>ex2. Lỗi khi lưu kết quả, hệ thống báo lỗi và cho phép thử lại.</p> <p>ex3. Sự cố kết nối hoặc gián đoạn trong quá trình làm bài, hệ thống có cơ chế lưu tạm thời hoặc hướng dẫn xử lý.</p>

Bảng 23: Usecase «Tạo báo cáo thống kê»

Usecase	Tạo báo cáo thống kê
Nguồn	L-US03, T-US04, NFR05, NFR06, NFR18
Use-case ID	UC-CL-05
Tổng quan	Trưởng khoa, giảng viên hoặc trợ giảng tạo báo cáo thống kê tiến độ và kết quả học tập sinh viên.
Actor	Dean / Lecturer / TA
Tiền điều kiện	<ul style="list-style-type: none"> Người dùng có quyền truy cập chức năng tạo báo cáo. Dữ liệu học tập đã được tổng hợp đầy đủ.

Usecase	Tạo báo cáo thống kê
Điều kiện kích hoạt	Người dùng chọn tạo báo cáo thống kê trong khóa học.
Các bước	<ol style="list-style-type: none">Chọn phạm vi dữ liệu và khoảng thời gian cần báo cáo.Chọn mẫu báo cáo (layout) nếu có.Hệ thống sinh báo cáo thống kê theo cấu hình.Người dùng xem, tải hoặc xuất báo cáo.
Hậu điều kiện	<ul style="list-style-type: none">Báo cáo được tạo đúng định dạng, dữ liệu chính xác.
Xử lý ngoại lệ	<p>ex1. Không có dữ liệu phù hợp để tạo báo cáo, hệ thống thông báo.</p> <p>ex2. Lỗi khi sinh báo cáo, hệ thống thông báo lỗi.</p> <p>ex3. Người dùng không có quyền truy cập chức năng, hệ thống từ chối.</p>

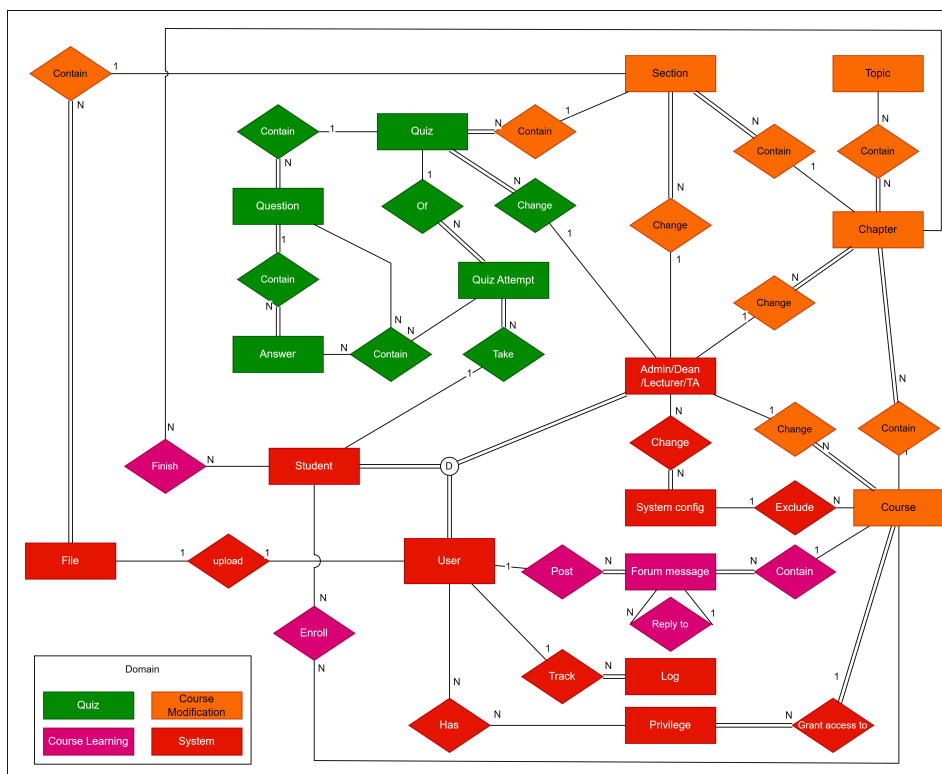
Bảng 24: Usecase «Tham gia diễn đàn khóa học»

Usecase	Tham gia diễn đàn khóa học
Nguồn	S-US03, T-US03, L-US04
Use-case ID	UC-CL-06
Tổng quan	Người dùng (sinh viên, giảng viên, trợ giảng) tham gia trao đổi thảo luận trên diễn đàn khóa học.
Actor	User
Tiền điều kiện	<ul style="list-style-type: none">Người dùng đã đăng nhập.Có quyền truy cập diễn đàn của khóa học.
Điều kiện kích hoạt	Người dùng truy cập và tham gia diễn đàn khóa học.
Các bước	<ol style="list-style-type: none">Xem các chủ đề và bài viết hiện có.Đăng bài mới hoặc trả lời các bài viết.Hệ thống lưu bài viết và cập nhật diễn đàn.
Hậu điều kiện	<ul style="list-style-type: none">Nội dung thảo luận được cập nhật, hỗ trợ trao đổi hiệu quả.
Xử lý ngoại lệ	<p>ex1. Lỗi khi lưu bài viết, hệ thống thông báo và đề nghị thử lại.</p> <p>ex2. Người dùng không có quyền truy cập, hệ thống từ chối và thông báo.</p> <p>ex3. Nội dung vi phạm quy định, hệ thống cảnh báo hoặc khóa bài.</p>

5 Phân tích & Thiết kế hệ thống

5.1 Entity Relationship Diagram (ERD)

5.1.1 Tổng quan



Hình 13: Entity Relationship Diagram

Tổng quan các Thực thể

Sơ đồ có tổng cộng 16 thực thể được chia thành 4 lĩnh vực (Domain): Quiz (Bài kiểm tra), Course Modification (Điều chỉnh Khóa học), Course Learning (Học tập Khóa học), và System (Hệ thống).

Course Modification, Quiz và Course Learning chia hệ thống thành 2 mảng tách rời nhưng tương trợ lẫn nhau: chỉnh sửa nội dung khóa học và sử dụng khóa học.

• Domain: Quiz (Bài kiểm tra)

Bao gồm các thực thể phục vụ định nghĩa các bài quiz, và ghi lại tương tác của Student trên các bài quiz này.

- **Quiz:** Đại diện cho một bài kiểm tra hoặc bài tập tổng thể.
- **Question:** Đại diện cho một câu hỏi cụ thể trong Quiz.
- **Answer:** Đại diện cho một lựa chọn trả lời khả dụng cho một Question.
- **Quiz Attempt:** Đại diện cho một lần làm Quiz cụ thể của một Student.

• Domain: Course Modification (Chỉnh sửa Khóa học)

Bao gồm các thực thể phục vụ định nghĩa các khóa học, phân chia một khóa học thành các đơn vị nhỏ, để truy xuất, tham chiếu.

- **Course:** Đại diện cho một khóa học hoàn chỉnh.
- **Chapter:** Đại diện cho một chương (bài học lớn) trong Course.
- **Topic:** Đại diện cho một chủ đề nhỏ hơn nằm trong Chapter.
- **Section:** Đại diện cho một phần nội dung, chứa các Quiz hoặc Topic.

• Domain: Course Learning (Học Khóa học)

Bao gồm các thực thể phục vụ các tương tác lên khóa học và quiz.

- **Forum message:** Đại diện cho một bài viết/bình luận trong diễn đàn.

• **Domain: System (Hệ thống)**

Bao gồm các thực thể phục vụ việc định danh người dùng, phân quyền, điều khiển và vận hành cho Admin.

- **User:** Đại diện cho người dùng chung của hệ thống (thực thể cha).
- **Admin / Dean / Lecturer / TA:** Đại diện cho người dùng có vai trò quản trị/giảng dạy, là thực thể con của User.
- **Student:** Đại diện cho người dùng có vai trò là học viên, là thực thể con của User.
- **File:** Đại diện cho các tài liệu được Upload lên hệ thống.
- **System config:** Đại diện cho các cài đặt cấu hình của toàn bộ hệ thống.
- **Privilege:** Đại diện cho các quyền hạn cụ thể của User.
- **Log:** Đại diện cho các bản ghi nhật ký hoạt động của User.

Các mối quan hệ giữa các thực thể

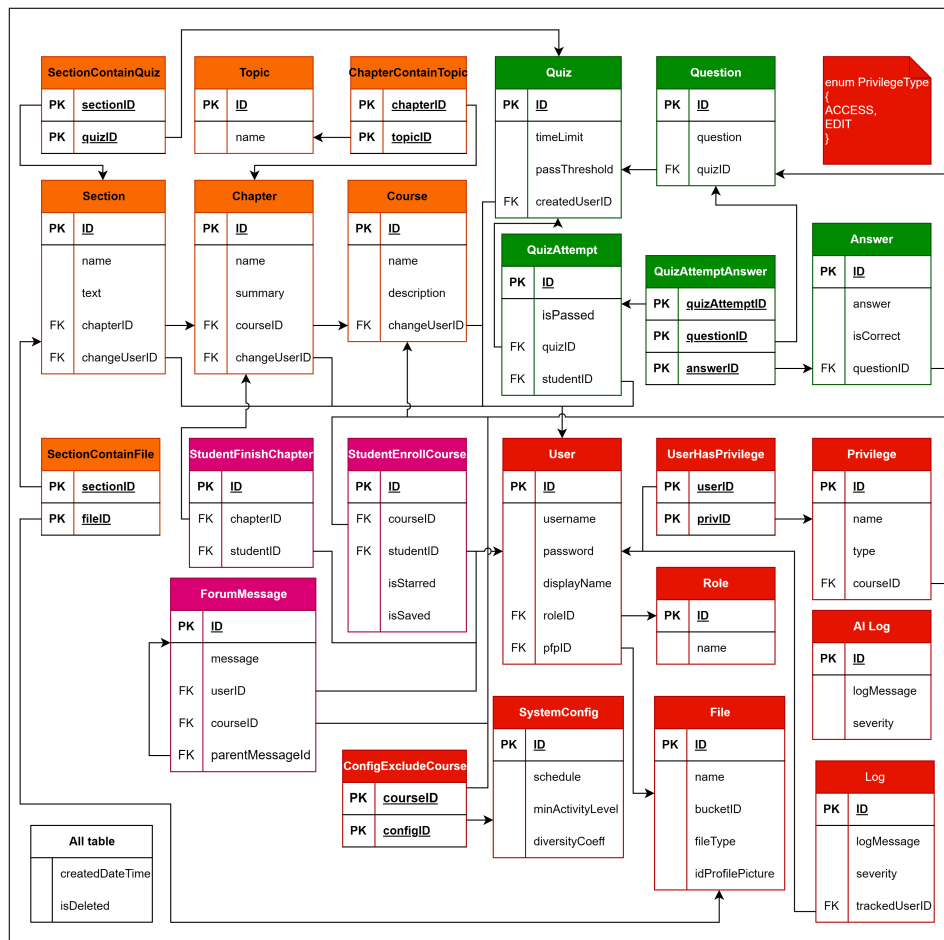
Domain	Thực thể 1	Tên Mối Quan hệ	Thực thể 2	Kiểu Mối Quan hệ (Cardinality)	Yêu cầu liên quan
Quiz	Quiz	Contain	Question	1:N - Một Quiz có nhiều Question	L-US02, T-US02, NFR07, NFR15
Quiz	Question	Contain	Answer	1:N - Một Question có nhiều Answer	
Quiz	Quiz Attempt	Of	Quiz	N:1 - Nhiều QuizAttempt thuộc về một Quiz	
Quiz	Student	Take	Quiz Attempt	1:N - Một Student thực hiện nhiều bài Quiz	
Quiz	Admin / Dean / Lecturer / TA	Change	Quiz	1:N - Một người dùng có quyền hạn có thể chỉnh sửa nhiều bài quiz	
Course Mod.	Course	Contain	Chapter	1:N - Một Course có nhiều Chapter	
Course Mod.	Chapter	Contain	Section	1:N - Một Chapter có nhiều Section	
Course Mod.	Chapter	Contain	Topic	1:N - Một Chapter có nhiều Topic	
Course Mod.	Section	Contain	File	1:N - Một Section có nhiều file (tài liệu, video,...)	
Course Mod.	Section	Contain	Quiz	1:N - Một Section có nhiều Quiz	
Course Mod.	Admin / Dean / Lecturer / TA	Change	Course	1:N - Một người dùng có quyền hạn chỉnh sửa nhiều Course	
Course Mod.	Admin / Dean / Lecturer / TA	Change	Chapter	1:N - Một người dùng có quyền hạn chỉnh sửa nhiều Chapter	
Course Mod.	Admin / Dean / Lecturer / TA	Change	Section	1:N - Một người dùng có quyền hạn chỉnh sửa nhiều Section	
Course Learn.	Student	Finish	Chapter	N:N - Nhiều Student hoàn thành nhiều Chapter	
Course Learn.	Student	Enroll	Course	N:N - Nhiều Student đăng ký nhiều Course	
Course Learn.	User	Post	Forum message	1:N - Một User đăng tải nhiều Forum message	
Course Learn.	User	Post	Forum message	1:N - Một User đăng tải nhiều Forum message	

Course Learn.	Course	Contain	Forum message	1:N - Một Course có nhiều Forum message	
System	User	Upload	File	1:1 - Một User đăng tải một File (ảnh đại diện)	
System	User	Has	Privilege	N:N - Nhiều User có nhiều Privilege	
System	Privilege	Grant access to	Course	N:1 - Nhiều Privilege cho phép truy cập một Course (2 dạng chỉ xem hoặc chỉnh sửa)	
System	Log	Track	User	N:1 - Nhiều Log theo dõi một User - để truy vấn trách nhiệm khi có sửa đổi	
System	Admin / Dean / Lecturer / TA	Change	System config	N:N - Nhiều người dùng có quyền hạn thay đổi nhiều System config	
System	System config	Exclude	Course	1:N - Một System config chứa thông tin về bỏ qua nhiều Course (hỗ trợ AI Service - use-case UC-SM-01)	

Mối Quan hệ Kế thừa (Inheritance/Specialization): Mỗi quan hệ này được biểu thị bằng ký hiệu vòng tròn **D** (Disjoint), nghĩa là người dùng Student sẽ không thể có được các quyền hạn ngang hàng với Admin, Lecturer, TA:

Mối Quan hệ bậc 3: Mỗi quan hệ Contain N-N-N giữa Quiz Attempt, Question và Answer có vai trò lưu giữ câu trả lời của Student cho mỗi câu hỏi tương ứng trong một lần làm quiz.

5.1.2 Mapping



Hình 14: Relational Mapping Diagram

Triển khai từ ERD ở phần trước, sơ đồ mapping bổ sung thêm các trường dữ liệu cho các bảng để hoàn thiện thiết kế cấu trúc dữ liệu.

Một số chi tiết đáng lưu ý:

- Bảng AI Log chỉ dùng để lưu trữ các thông tin hoạt động của AI service, không có quan hệ với bảng nào khác.
- Tất cả các bảng đều có các trường createDateTime - thời điểm record đó được tạo ra, và isDeleted - hỗ trợ tính năng soft-delete
- Privilege được chia làm 2 loại (type): ACCESS - chỉ truy cập và đọc nội dung, dành cho Student, EDIT - truy cập và chỉnh sửa nội dung, dành cho Lecturer, TA.

5.2 Kiến trúc hệ thống

5.2.1 Đặc tính kiến trúc (Architecture characteristics)

Ứng dụng phương pháp **Architecture Katas** lên các yêu cầu hệ thống ở bảng 1 và 2, nhóm đề xuất ra 3 đặc tính chính của hệ thống:

1. **Configurability**, vì:

- Quản lý hệ thống**: Cho phép thay đổi cách thức vận hành của hệ thống mà không cần can thiệp hoặc chỉnh sửa trực tiếp vào mã nguồn. Đặc tính này bao gồm việc *Đối với mô hình Trí tuệ nhân tạo (AI)* thì có khả năng thiết lập *chu kỳ tự động bảo trì*. Tương ứng với: NFR01, NFR02.
- Chỉnh sửa khóa học**: Cho phép cấu hình *kiểu tài liệu* người dùng được phép đăng tải, cũng như chỉnh sửa *layout (bố cục) giao diện khóa học* hiển thị cho người dùng. Tương ứng với: NFR03, NFR04.

- **Tạo báo cáo:** Người dùng có thể *thay đổi layout* của báo cáo được tạo ra và *thay đổi phạm vi dữ liệu* được sử dụng để lập báo cáo. Tương ứng với: NFR05, NFR06.
 - **Tạo câu hỏi bằng AI:** Cho phép cấu hình các tham số tạo câu hỏi như *số lượng câu*, *độ khó* mong muốn, và *phân bố độ khó* giữa các câu hỏi. Tương ứng với: NFR07.
 - **Hỏi gợi ý:** Có thể thiết lập *tần suất* hiển thị gợi ý và *mức độ chi tiết* của gợi ý. Tương ứng với: NFR08.
 - **Đề xuất chủ đề liên quan:** Cho phép người dùng *thêm* các chủ đề quan tâm hoặc *bỏ* các chủ đề không quan tâm ra khỏi danh sách đề xuất. Tương ứng với: NFR09.
2. **Availability** : Đặc tính này tập trung vào việc đảm bảo hệ thống luôn có thể được truy cập và sử dụng, đặc biệt là trong các tình huống *thời gian sinh viên truy cập không ổn định* hoặc có sự tập trung người dùng lớn. Việc mất khả năng truy cập hệ thống sẽ ảnh hưởng đến chất lượng học tập của sinh viên - lớp người dùng luôn phải cân đối thời gian cho mọi việc, cũng như ảnh hưởng tiến độ xây dựng bài học của các giảng viên, trợ giảng,... Tương ứng với: NFR10, NFR11, NFR12.

3. Data consistency + integrity , vì:

- Yêu cầu hệ thống phải *đảm bảo tính đúng đắn* của các tài liệu bài giảng được đăng tải. Tương ứng với: NFR13.
- Yêu cầu *dữ liệu đăng tải phải được giữ an toàn* khỏi các truy cập trái phép hoặc mất mát. Tương ứng với: NFR14.
- Đảm bảo *mọi dữ liệu hiển thị phải đồng bộ* và nhất quán giữa tất cả các người dùng đang sử dụng hệ thống. Tương ứng với: NFR15.
- Các thao tác quan trọng phải đảm bảo tính toàn vẹn giao dịch (transactional integrity) và lịch sử chỉnh sửa nội dung khóa học phải được ghi lại để phục vụ truy vết. Tương ứng với: NFR16, NFR17

Các đặc tính phụ liên quan

Các đặc tính sau đây mô tả các yêu cầu hiệu năng và khả năng mở rộng của hệ thống:

- **Performance (Hiệu năng):** Yêu cầu *tối ưu hóa hiệu năng* của các mô hình AI để đảm bảo phản hồi nhanh chóng. Tương ứng với: NFR18, NFR19, NFR20.
- **Concurrency (Đồng thời):** Đảm bảo hệ thống có khả năng xử lý và phục vụ cho *nhiều người dùng cùng lúc* mà không gặp lỗi hoặc giảm chất lượng dịch vụ. Tương ứng với: NFR21, NFR22.
- **Elasticity (Đàn hồi/Co giãn):** Đảm bảo hệ thống có khả năng mở rộng hoặc thu hẹp tài nguyên để đối phó với *thời gian sử dụng không đều*, đặc biệt là khi tải trọng hệ thống *tập trung cao vào mùa thi*. Tương ứng với: NFR23, NFR24, NFR25.

5.2.2 So sánh các phong cách kiến trúc khả thi

Bảng 26: So sánh giữa Service-Based Architecture (SBA) và Microservices Architecture (MSA)

Tính chất	Service-Based Architecture (SBA)	Microservices Architecture (MSA)
Định nghĩa Dịch vụ	Các dịch vụ lớn, phân tách theo lĩnh vực nghiệp vụ (Domain).	Các dịch vụ rất nhỏ, phân tách theo chức năng hoặc thực thể (Entity).
Độc lập Triển khai	Có thể độc lập triển khai, nhưng thường liên quan nhau.	Hoàn toàn độc lập triển khai.
Quản lý Dữ liệu	Dễ dàng hơn; có thể chia sẻ CSDL vật lý (tách schema).	Phức tạp hơn; mỗi dịch vụ có CSDL riêng (Data isolation).
Tính nhất quán Dữ liệu (NFR15, NFR16)	Dễ đạt được (Giao dịch cục bộ).	Khó khăn (Cần giao dịch phân tán/Saga).
Khả năng Mở rộng	Tốt ở cấp độ dịch vụ (Service Level).	Tốt nhất ở cấp độ chức năng (Function Level).
Đơn giản Vận hành	Cao hơn (ít dịch vụ hơn).	Thấp hơn (phức tạp về mạng, giám sát).
Chi phí Vận hành	Thấp hơn.	Cao hơn (Do cần nhiều tài nguyên và công cụ).

Tính phù hợp	Hệ thống phức tạp trung bình, cần cân bằng giữa phân tách và quản lý.	Hệ thống quy mô lớn, cần mở rộng tối đa và đa dạng công nghệ.
---------------------	---	---

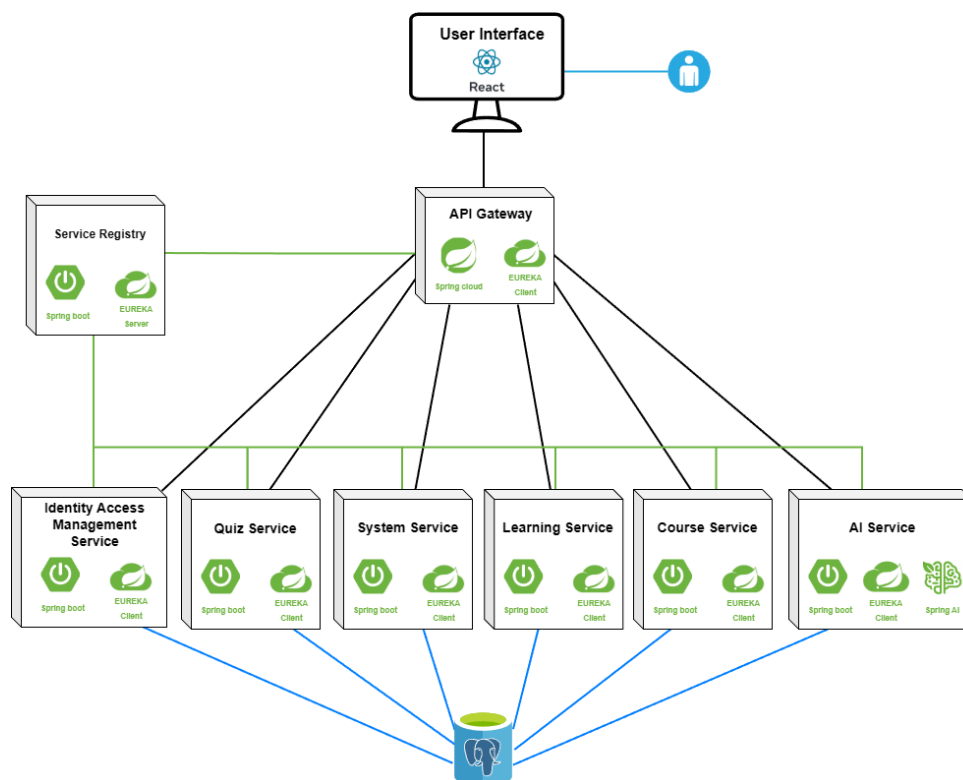
Quyết định: Service-Based Architecture (SBA) được chọn làm phong cách kiến trúc hiện tại.

Lý do:

- SBA cung cấp sự phân tách nghiệp vụ cần thiết để đáp ứng tốt các yêu cầu về **Configurability** (NFR01–NFR09) và **Elasticity** (NFR23).
- SBA giảm thiểu rủi ro phức tạp trong việc quản lý **Tính nhất quán dữ liệu** (NFR15, NFR16) so với MSA, nơi mà việc này đòi hỏi giao dịch phân tán phức tạp.
- Theo nguyên tắc **YAGNI** và **DRY**, độ phức tạp của MSA là không cần thiết cho quy mô hiện tại của hệ thống.
- Chi tiết xem tại mục **9.1.2**

5.2.3 Thiết kế kiến trúc

5.2.3.a Module view



Hình 15: Module view - Nguyên tắc tổ chức

1. Nguyên tắc Tổ chức Module

- Hệ thống được thiết kế dựa trên **Service-Based Architecture (SBA)** nhằm tối ưu hóa tính module hóa và đảm bảo **phân tách mối quan tâm (separation of concerns)**.
- Cấu trúc Module View chi tiết hóa cách thức mã nguồn hệ thống được tổ chức thành 9 module, tuân thủ **Nguyên tắc Trách nhiệm Đơn nhất (Single Responsibility Principle)**.

2. Mô tả Tổng quát các Module

Các module được phân loại thành hai nhóm chính: Dịch vụ Nghiệp vụ Miền (Domain Services) và Dịch vụ Hỗ trợ Hạ tầng và Truy cập (Infrastructure & Access Services).

(a) Dịch vụ Nghiệp vụ Miền (Domain Services)

Gồm 6 module cốt lõi, mỗi module tập trung quản lý một miền nghiệp vụ chuyên biệt:

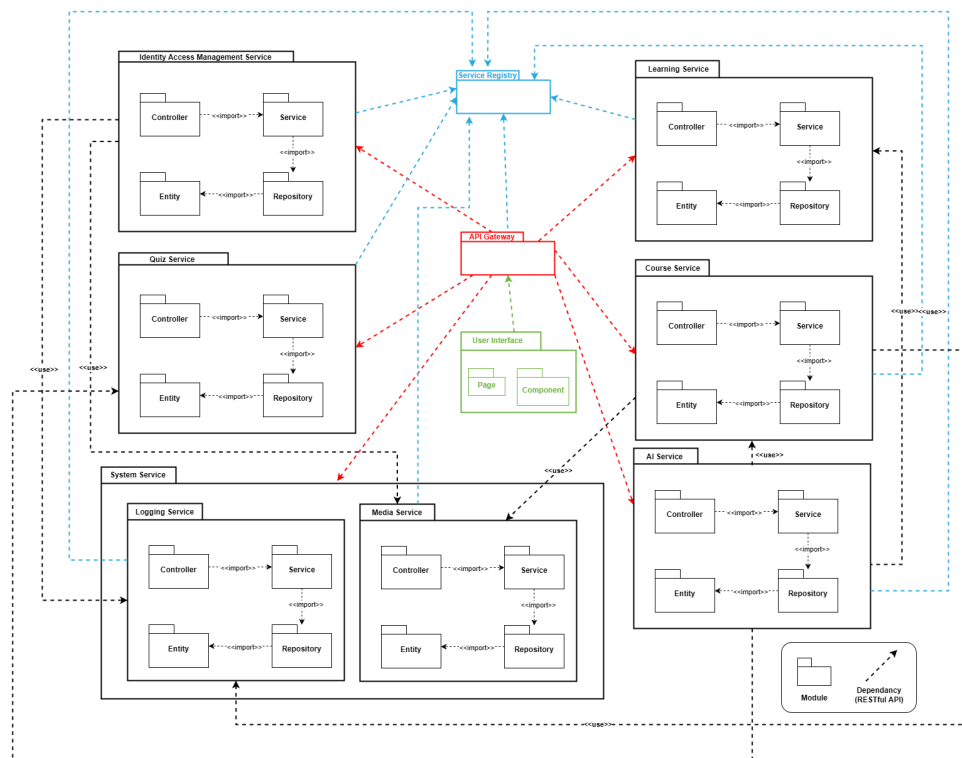
Module	Mục đích	Trách nhiệm chính
IAM Service	Quản lý Danh tính, Truy cập và Thông tin người dùng.	Thực hiện xác thực và phân quyền; quản lý hồ sơ, quyền hạn, vai trò và vòng đời người dùng.
Quiz Service	Quản lý mọi hoạt động liên quan tới bài kiểm tra.	Tạo, chỉnh sửa, lưu trữ cấu trúc câu hỏi và bài quiz; chấm điểm và lưu trữ phiên làm bài.
System Service	Cung cấp các chức năng hỗ trợ chung mang tính nền tảng.	Cung cấp cơ chế logging cho các thao tác bảo mật; quản lý các tài nguyên media (video, document, ảnh).
Learning Service	Quản lý hành trình học tập và tiến trình của người dùng.	Theo dõi tiến trình học (completion, progress); quản lý các forum/thảo luận của mỗi khóa học.
Course Service	Quản lý vòng đời của khóa học.	Quản lý metadata, trạng thái khóa học; xử lý quy trình đăng ký học viên và các quy tắc truy cập.
AI Service	Hỗ trợ học viên thông qua việc áp dụng Trí tuệ Nhân tạo.	Phân tích hành vi và tiến trình học; đề xuất nội dung, tài liệu, hoặc lộ trình học tập phù hợp.

(b) Dịch vụ Hỗ trợ Hạ tầng và Truy cập (Infrastructure & Access Services)

Các module thiết yếu trong việc quản lý kết nối, giao tiếp và cấu hình:

- **API Gateway:** Đóng vai trò là Điểm truy cập duy nhất (Single Entry Point) cho client. Trách nhiệm chính là Định tuyến yêu cầu tới dịch vụ tương ứng.
- **Service Registry:** Cung cấp cơ chế Tìm kiếm Dịch vụ (Service Discovery). Trách nhiệm chính là Lưu trữ metadata và danh sách các instance đang chạy của các dịch vụ.
- **User Interface (UI):** Là điểm tương tác cuối cùng với người dùng. Trách nhiệm chính là Hiển thị nội dung, thu thập input, gọi các API qua API Gateway và xử lý kết quả.

3. Tổ chức, phân lớp, phụ thuộc giữa các Module

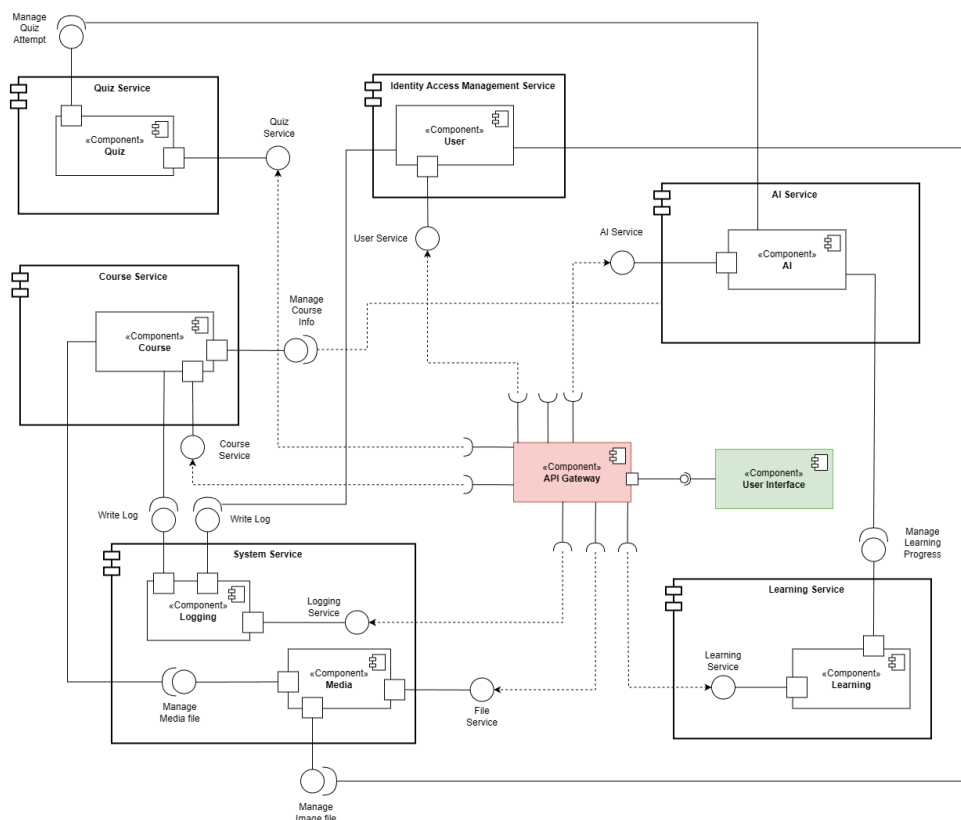


Hình 16: Module view - Package diagram

- (a) **Nguyên tắc Phân lớp Nội bộ:** Mỗi module dịch vụ đều tuân theo **Layered Architecture** nội bộ thống nhất (Controller → Service → Repository → Entity) để phân tách rõ ràng trách nhiệm.

- (b) **Luồng Truy cập:** UI → API Gateway → Domain Services. Gateway đóng vai trò là **Facade**, bảo vệ các dịch vụ nghiệp vụ.
- (c) **Phụ Thuộc giữa các Module (Inter-Module Dependencies):** Các module giao tiếp qua REST API theo hai kiểu:
- **Phụ thuộc qua API Gateway («use»):** Tất cả Domain Services đều có mối quan hệ «use» với API Gateway (Gateway là thành phần duy nhất được phép gọi các dịch vụ này).
 - **Phụ thuộc giữa các Domain Services (S2S):**
 - IAM Service và Course Service đều «use» **Media Service** và **Logging Service** (trong System Service).
 - AI Service «use» **Learning Service**, **Course Service**, **Quiz Service** để lấy dữ liệu tiến trình học và điểm số.
- (d) **Phụ thuộc vào Service Registry:** Tất cả Domain Service và API Gateway phụ thuộc vào Service Registry theo hai chiều:
- **Đăng ký (Registration):**
 - Mỗi Domain Service (IAM, Learning, Quiz, Course, System, AI) khi khởi động sẽ tự động đăng ký các instance của mình bao gồm địa chỉ IP, cổng và các metadata cần thiết.
 - Service Registry duy trì danh sách các instance hợp lệ dựa trên cơ chế Heartbeat/Healthcheck, TTL (Time-To-Live), và thực hiện deregister khi một service không còn hoạt động. Việc đăng ký là bắt buộc, vì nếu một service không được đăng ký, các service khác sẽ không thể phát hiện và gọi đến nó.
 - **Tra cứu (Discovery):**
 - Các thành phần cần gọi tới dịch vụ khác sẽ thực hiện tra cứu thông qua Service Registry để lấy thông tin địa chỉ runtime của các service. Ví dụ, API Gateway khi nhận request từ client sẽ query Service Registry để xác định instance phù hợp của service tương ứng dựa trên tên logic.
 - Tương tự, trong các kịch bản service-to-service, một service khi muốn gọi service khác (ví dụ AI tới Learning) cũng phải tham khảo Registry để lấy endpoint mới nhất.
- Cơ chế này cho phép giao tiếp giữa các service dựa trên tên service thay vì địa chỉ IP hoặc cổng cứng, từ đó đảm bảo loose coupling giữa các thành phần trong hệ thống.

5.2.3.b Component & Connector View



Hình 17: Component & Connector view

1. Identity & Access Management (IAM) Service

- **User Component:** Xử lý toàn bộ tác vụ liên quan đến danh tính và truy cập người dùng: đăng ký, đăng nhập, xác thực (authentication), quản lý phiên, cấp token và quản lý hồ sơ người dùng.
- **Interface:** User Service — cung cấp cho API Gateway để thực hiện xác thực và truy vấn thông tin người dùng.

2. Quiz Service

- **Quiz Component:** Quản lý toàn bộ nghiệp vụ về bài kiểm tra: tạo quiz, quản lý câu hỏi, thực hiện quiz, chấm điểm tự động và lưu kết quả. Tích hợp với Learning Service để cung cấp dữ liệu đánh giá.
- **Manage Quiz Attempt (Interface):** Cung cấp cho AI Service để lấy thông tin về kết quả các lần làm quiz của học viên trong khóa học, từ đó đưa ra gợi ý lộ trình học.
- **Interface:** Quiz Service — cho phép API Gateway gửi request về quiz và kết quả đánh giá.

3. Course Service

- **Course Component:** Quản lý nội dung khóa học: tạo khóa học, chương, bài học; quản lý nội dung đa phương tiện; thiết lập điều kiện tiên quyết và quyền truy cập. Tương tác với Media Component để lưu trữ tài nguyên.
- **Manage Course Info (Interface):** Cung cấp cho AI Service để lấy thông tin về khóa học của học viên.
- **Interface:** Course Service — dùng bởi API Gateway để thao tác dữ liệu khóa học.

4. Learning Service

- **Learning Component:** Theo dõi tiến độ học tập: ghi nhận trạng thái hoàn thành, thời gian học, điểm số, tính phần trăm tiến độ và lưu lịch sử học. Cung cấp dữ liệu cho AI Service để phân tích và gợi ý.
- **Manage Learning Progress (Interface):** Cung cấp cho AI Service để lấy thông tin về quá trình học của học viên.

- **Interface: Learning Service** — cho API Gateway truy vấn tiến độ và lịch sử học tập.

5. AI Service

- **AI Component:** Phân tích dữ liệu từ Learning, Course và Quiz để xây dựng mô hình học tập cá nhân hóa; đề xuất lộ trình học, khóa học phù hợp và gợi ý nội dung thông minh.
- **Interface: AI Service** — cung cấp cho API Gateway để lấy đề xuất cá nhân hóa.

6. System Service

- **Logging Component:** Thu thập và lưu trữ log từ mọi service; phân loại log; hỗ trợ phân tích và giám sát hệ thống.
 - **Write Log (Interface):** Cung cấp cho IAM Service và Course Service để ghi log lại các thao tác về xác thực, đăng ký người dùng và tạo, chỉnh sửa khóa học.
 - **Interface: Logging Service** — cung cấp cho API Gateway để xem log.
- **Media Component:** Xử lý tài nguyên đa phương tiện: upload, lưu trữ, tối ưu hóa và phân phối nội dung media. Hỗ trợ Course và IAM (ảnh đại diện).
 - **Manage media file, Manage image file (Interfaces):** Cung cấp cho IAM Service và Course Service để thao tác với các file media như ảnh đại diện, video bài giảng.
 - **Interface: File Service** — cho API Gateway thao tác với file media.

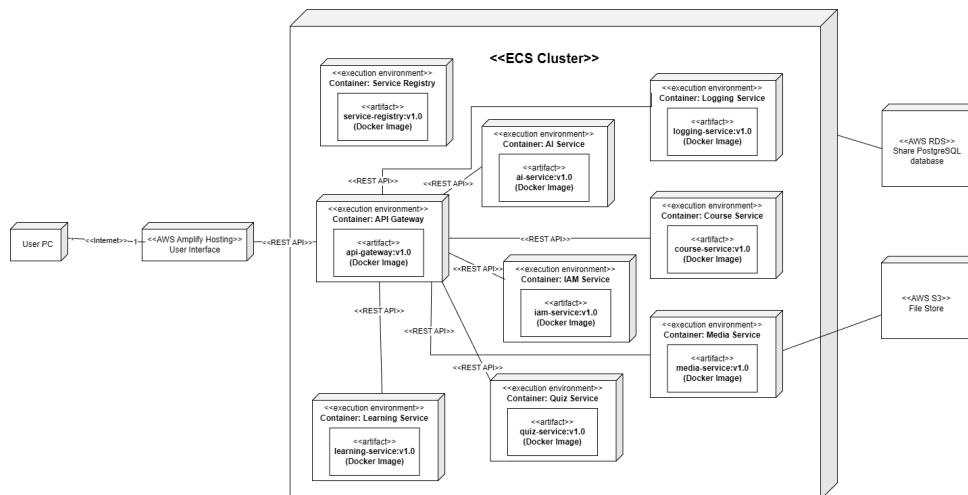
7. API Gateway Component

- **Chức năng:** Thực hiện định tuyến request tới đúng service, xác thực/ủy quyền dựa trên IAM, cân bằng tải và áp dụng chính sách bảo mật tập trung.

8. User Interface Component

- **Chức năng:** Tầng giao diện người dùng: hiển thị nội dung khóa học, làm quiz, xem tiến độ, nhận gợi ý từ AI và quản lý hồ sơ cá nhân.
- **Giao tiếp:** Giao tiếp với backend thông qua API Gateway.

5.2.3.c Allocation view



Hình 18: Component & Connector view

1. Tổng quan mô hình triển khai hệ thống

Hệ thống được triển khai theo mô hình service-based architecture trên nền tảng container hóa, trong đó toàn bộ các service backend được triển khai tập trung trong một ECS Cluster thuộc Amazon Web Services. Người dùng truy cập hệ thống từ máy cá nhân thông qua Internet, giao diện người dùng được host độc lập trên AWS Amplify. Tất cả các yêu cầu từ phía frontend đều được chuyển tiếp về backend thông qua API Gateway, đóng vai trò là điểm vào duy nhất của hệ thống. Mô hình này giúp tách biệt rõ ràng giữa tầng giao diện và tầng xử lý nghiệp vụ, đồng thời tăng cường tính bảo mật và khả năng kiểm soát truy cập.

2. Kiến trúc service-based trong ECS Cluster

Trong ECS Cluster, mỗi service được triển khai dưới dạng một container độc lập, tương ứng với một Docker image (artifact). Hệ thống không sử dụng kiến trúc microservices mà áp dụng mô hình service-based, trong đó các service vẫn tách biệt về mặt triển khai, chức năng và vòng đời, nhưng chia sẻ cùng một cơ sở dữ liệu trung tâm. Các service chính bao gồm: API Gateway, Service Registry, IAM Service, AI Service, Course Service, Learning Service, Quiz Service, Media Service và Logging Service. Mỗi service tồn tại như một môi trường thực thi riêng biệt (execution environment) và giao tiếp với các service khác thông qua REST API trên mạng nội bộ của ECS.

3. Cơ chế định tuyến và cân bằng tải với Spring Cloud Gateway và Eureka

API Gateway trong mô hình này được hiện thực bằng Spring Cloud Gateway và tích hợp với Eureka từ Service Registry. Thông qua cơ chế service discovery, API Gateway có khả năng tự động phát hiện các instance của service, thực hiện cân bằng tải (load balancing), kiểm tra trạng thái (health check) và định tuyến request đến đúng service còn hoạt động. Điều này giúp hệ thống đạt được tính chịu lỗi cao (fault tolerance) và khả năng mở rộng động (dynamic scalability) khi số lượng instance của từng service thay đổi.

4. Service Registry và cơ chế phát hiện dịch vụ

Service Registry được triển khai như một container riêng trong ECS Cluster và đóng vai trò trung tâm trong việc đăng ký và tra cứu service. Khi mỗi service khởi động, nó sẽ tự động đăng ký thông tin (tên service, địa chỉ, cổng, trạng thái) với Registry. API Gateway dựa trên thông tin này để định tuyến request một cách linh hoạt mà không cần cấu hình cứng địa chỉ từng service. Kiến trúc này giúp hệ thống tránh phụ thuộc tĩnh, đồng thời đặc biệt phù hợp với môi trường container có khả năng scale linh động theo tải.

5. Kiến trúc tầng dữ liệu và lưu trữ

Về tầng dữ liệu, hệ thống sử dụng một PostgreSQL Database dùng chung được triển khai dưới dạng dịch vụ managed của Amazon RDS for PostgreSQL. Tất cả các service nghiệp vụ, bao gồm IAM Service, AI Service, Course Service, Learning Service, Quiz Service và Logging Service, đều kết nối trực tiếp tới cơ sở dữ liệu này để đọc và ghi dữ liệu. Riêng Media Service không lưu trữ dữ liệu file trong database mà chỉ sử dụng database để quản lý metadata, còn toàn bộ dữ liệu media vật lý được lưu trữ trên Amazon S3. Việc tách riêng dữ liệu quan hệ và dữ liệu file giúp tối ưu hiệu năng truy xuất, chi phí lưu trữ và khả năng mở rộng hệ thống.

6. Quy trình triển khai hệ thống và CI/CD với GitHub Actions và CloudFormation

Quy trình triển khai của hệ thống được tự động hóa theo mô hình CI/CD. Mỗi service được build từ mã nguồn và đóng gói thành một Docker image với phiên bản tương ứng, chẳng hạn như api-gateway:v1.0, course-service:v1.0, media-service:v1.0. Khi push source code lên repository trên GitHub, GitHub Actions sẽ tự động kích hoạt pipeline CI để thực hiện các bước: kiểm tra mã nguồn, build Docker image và đẩy image lên Amazon Elastic Container Registry (ECR).

Sau khi image được lưu trữ trên ECR, quá trình triển khai hạ tầng và cập nhật service được thực hiện thông qua AWS CloudFormation. CloudFormation chịu trách nhiệm khởi tạo và cập nhật các tài nguyên như ECS Cluster, Task Definition, Service, Load Balancer và các cấu hình mạng liên quan. ECS sẽ tự động pull các image mới nhất từ ECR để khởi tạo hoặc cập nhật container. Khi container khởi động, các service sẽ kết nối tới Service Registry để đăng ký, đồng thời thiết lập kết nối tới PostgreSQL Database, riêng Media Service thiết lập thêm kết nối tới S3. Sau khi toàn bộ service sẵn sàng, API Gateway bắt đầu tiếp nhận request từ frontend và điều phối xử lý trong nội bộ hệ thống. Quy trình này cho phép triển khai tự động, lặp lại, giảm sai sót thủ công và rút ngắn thời gian release.

7. Luồng xử lý tổng thể trong môi trường triển khai

Luồng xử lý tổng thể của hệ thống diễn ra theo trình tự: người dùng thao tác trên giao diện web được host trên AWS Amplify, request được gửi qua Internet tới API Gateway trong ECS Cluster. API Gateway tiến hành xác thực thông qua IAM Service, sau đó dựa trên Service Registry để định tuyến request tới Course Service, Learning Service, Quiz Service, AI Service hoặc Media Service tùy theo nghiệp vụ. Các service này xử lý logic chuyên biệt và truy cập trực tiếp vào PostgreSQL Database để lấy hoặc cập nhật dữ liệu. Trong trường hợp xử lý file, Media Service sẽ thực hiện lưu trữ và truy xuất nội dung từ S3. Toàn bộ hoạt động của hệ thống được Logging Service ghi nhận và lưu trữ phục vụ công tác giám sát, theo dõi và phân tích hệ thống.

6 Hiện thực

7 Triển khai

8 Tổng kết

8.1 Nhận xét

8.2 Hướng phát triển

9 Phụ lục

9.1 ADR

9.1.1 ADR 00: Lựa chọn kiến trúc ban đầu: Microservices Architecture (MSA)

1. **Tiêu đề:** Lựa chọn kiến trúc ban đầu: Microservices Architecture (MSA).
2. **Status (Trạng thái):** Superseded (Đã bị thay thế).
3. **Context (Bối cảnh):**
 - (a) Hệ thống dự kiến sẽ phục vụ lượng lớn sinh viên, đặc biệt là vào các đợt cao điểm như mùa thi, đòi hỏi khả năng co giãn tối đa (Maximum Scalability).
 - (b) Cần tối ưu hóa hiệu năng cho các Module AI (NFR18, NFR25) bằng cách sử dụng các công nghệ khác nhau (Polyglot) cho từng thành phần.
 - (c) Các yêu cầu về **Elasticity** (NFR23, NFR24) và **Concurrency** (NFR21) là rất cao và cần được hỗ trợ bằng việc cô lập tài nguyên cho từng tính năng.
4. **Decision (Quyết định):**
 - (a) Áp dụng **Microservices Architecture (MSA)** để phân tách các Module chức năng thành các dịch vụ độc lập (ví dụ: Quiz Generation Service, User Service).
 - (b) Mỗi dịch vụ sẽ có khả năng **triển khai độc lập** và **mở rộng độc lập** (Autoscaling) dựa trên tải trọng CPU/GPU hoặc số lượng yêu cầu.
 - (c) Sử dụng **Eventual Consistency** cho các giao tiếp liên Service, chấp nhận độ trễ nhỏ để đạt được tính độc lập cao.
5. **Consequences (Hệ quả):**
 - (a) **Khả năng mở rộng tối đa (Maximum Elasticity):** Đáp ứng tốt yêu cầu NFR25 bằng cách mở rộng các dịch vụ AI một cách chi tiết.
 - (b) **Tăng độ phức tạp về vận hành (DevOps):** Yêu cầu đội ngũ vận hành mạnh và các công cụ giám sát phức tạp.
 - (c) **Rủi ro về Data Consistency:** Phải quản lý giao dịch phân tán (Saga) giữa các CSDL riêng biệt, làm tăng độ phức tạp trong việc đảm bảo NFR16.

9.1.2 ADR 01: Chuyển đổi kiến trúc từ Microservices (MSA) sang Service-Based Architecture (SBA)

1. **Tiêu đề:** Chuyển đổi kiến trúc từ Microservices (MSA) sang Service-Based Architecture (SBA).
2. **Status (Trạng thái):** Accepted (Đã chấp nhận).
3. **Context (Bối cảnh):**
 - (a) Hệ thống có yêu cầu cao về **Tính nhất quán giao dịch dữ liệu** (NFR16) và các mối quan hệ dữ liệu phức tạp giữa các Domain (Quiz, Course, Student).
 - (b) Độ phức tạp quản lý giao dịch phân tán (Saga) của MSA sẽ làm tăng chi phí và thời gian phát triển đáng kể.
 - (c) Mặc dù yêu cầu về tải có đỉnh điểm (mùa thi), quy mô hiện tại phù hợp hơn với kiến trúc ít phức tạp hơn.
4. **Decision (Quyết định):**
 - (a) Áp dụng **Service-Based Architecture (SBA)**, phân chia hệ thống thành các dịch vụ lớn (User/Access, Course Modification, Quiz/Assessment, Learning/Interaction, Reporting/Config).
 - (b) Sử dụng **API Gateway** để định tuyến, chứng thực và cung cấp lớp caching thống nhất (NFR20).
 - (c) Sử dụng **Service Registry** để quản lý vị trí động và trạng thái của các dịch vụ, hỗ trợ **Elasticity** (NFR23, NFR24) và **Availability** (NFR10, NFR11).
5. **Consequences (Hệ quả):**
 - (a) **Cải thiện Data Consistency (NFR15, NFR16):** Dễ dàng đạt được thông qua giao dịch cục bộ hoặc quản lý CSDL đơn giản hơn.
 - (b) **Giảm chi phí vận hành** và tăng tốc độ phát triển.
 - (c) Vẫn giữ được tính linh hoạt mở rộng nhờ các thành phần hỗ trợ (API Gateway, Service Registry).

10 Tài liệu tham khảo

11 Phụ lục