

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**----🙣🕮🙡----**

A logo with a letter h and a flame

Description automatically generated

BÁO CÁO MÔN HỌC:

KIẾN TRÚC PHẦN MỀM

*Đề tài:*

Mô tả yêu cầu cơ bản của một hệ thống quản lý lớp học tín chỉ

Giảng viên hướng dẫn : Võ Văn Hải

Nhóm : Nhóm 11

Lớp học phần : DHKTPM16DTT

***TP.HCM, ngày 20 tháng 5 năm 2024.***

**THÀNH VIÊN NHÓM**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và Tên | Mã Số Sinh Viên | Chữ Ký |
| 1 | Nguyễn Thanh Bảo Long | 20109191 |  |
| 2 | Nguyễn Hữu Thịnh | 20084921 |  |

Mục Lục:

[Chương 1: TỔNG QUÁT 4](#_Toc167107434)

[1. Đề bài 4](#_Toc167107435)

[2. Mô tả chi tiết lại đề bài 5](#_Toc167107436)

[3. Mục tiêu đề bài 6](#_Toc167107437)

[4. Mô tả yêu cầu chức năng 6](#_Toc167107438)

[5. Mô tả yêu cầu phi chức năng 7](#_Toc167107439)

[Chương 2 PHÂN TÍCH VÀ THIẾT KẾ 8](#_Toc167107440)

[1. Danh sách tác nhân và mô tả 8](#_Toc167107441)

[2. Sơ đồ class tổng quát 8](#_Toc167107442)

[Chương 3: KIẾN TRÚC 9](#_Toc167107443)

[1. Chọn kiến trúc: 9](#_Toc167107444)

[2. Lý do chọn kiến trúc 9](#_Toc167107445)

[3. Thiết kế 10](#_Toc167107446)

[4. Cấu trúc code 12](#_Toc167107447)

[5. So Sánh Với Những Kiến Trúc Khác 13](#_Toc167107448)

[Chương 4: TỔNG KẾT 15](#_Toc167107449)

[1. Tổng quan kiến trúc microservice: 15](#_Toc167107450)

[2. Kết quả đạt được 15](#_Toc167107451)

3. Những Hạn Chế …………………………………………………………… 15

[3. Hướng phát triển trong tương lai: 16](#_Toc167107452)

# Chương 1: TỔNG QUÁT

## Đề bài

Một trường đại học cần quản lý hệ thống đăng ký học phần của sinh viên theo theo cơ chế tín chỉ. Các sinh viên sẽ được biên chế ở các khoa theo mỗi ngành học của mình đăng ký.

Trong mỗi học kỳ, nhà trường sẽ chọn một số môn học của từng ngành học cho sinh viên đăng ký học. Sinh viên đăng ký theo học vào một lớp nhất định (lớp tín chỉ) với số sinh viên được đăng ký tối đa được qui định cho từng lớp.

Môn học được mở có thể là môn học tự chọn

Mỗi môn học có một hoặc nhiều môn học tiên quyết. Sinh viên sẽ không được đăng ký môn học mà có môn tiên quyết mình chưa học.

Mỗi học kỳ sinh viên được đăng ký tối đa 30 tín chỉ. Nếu quá số này, hệ thống sẽ không cho đăng ký.

Để phòng ngừa trường hợp sinh viên đăng ký xong rồi hủy bỏ, nhà trường yêu cầu sinh viên phải xác nhận trước khi đăng ký.

Vào ngày mở đăng ký, sinh viên sẽ đăng nhập vào hệ thống và sẽ nhìn thấy danh sách các môn học mà mình có khả năng đăng ký. Sinh viên chọn các môn học và tiến hành đăng ký. Trường hợp các lớp đã đầy, sinh viên sẽ được đưa vào một danh sách dự bị để nhà trường cân nhắc có mở thêm lớp hay không. Nếu không mở thêm lớp, sinh viên sẽ bị hủy đăng ký môn đó.

Sau khi đăng ký thành công, một email thông báo sẽ được gửi cho sinh viên xác nhận việc đăng ký và nhận quyết định đóng học phí.

Ngoài ra, hệ thống còn có khả năng cho phép sinh viên xem thông tin học tập của mình (số tín chỉ đã đạt, số môn đã học, điểm môn học, điểm trung bình tích lũy…), thời khóa biểu theo tuần và các tiện ích khác.

Sau khi số tín chỉ đã đạt theo từng ngành, sinh viên có quyền đăng ký xét tốt nghiệp. Nếu mọi tiêu chuẩn đều thỏa mãn, sinh viên sẽ được cấp bằng tốt nghiệp và sẽ được đưa vào danh sách các cựu sinh viên. Thông tin về bằng cấp sẽ được công khai trên trang web của nhà trường. Thông tin của cựu sinh viên sẽ được lưu giữ để theo dõi quá trình làm việc (nếu sinh viên đồng ý), làm các cuộc survey, cũng như nhiều hoạt động khác.

## Mô tả chi tiết lại đề bài

Trong việc phân chia sinh viên vào các khoa theo ngành học, trường đại học cần thiết kế một hệ thống quản lý học phần hiệu quả để đảm bảo rằng việc đăng ký diễn ra một cách trơn tru và minh bạch. Hệ thống này cần phải có khả năng tự động cập nhật thông tin về các môn học, bao gồm cả các môn học bắt buộc và tự chọn, cũng như các yêu cầu về môn tiên quyết.

Việc tổ chức các lớp học cũng là một phần quan trọng trong quá trình quản lý học phần. Trường cần xác định số lượng sinh viên tối đa cho mỗi lớp dựa trên yêu cầu giảng dạy và cơ sở vật chất. Ngoài ra, cần có sự linh hoạt để điều chỉnh số lượng lớp và lịch trình học tùy thuộc vào nhu cầu và sự thay đổi trong số lượng sinh viên đăng ký.

Hệ thống cũng cần tích hợp các tính năng như thông báo và cập nhật tự động để thông tin về các môn học, lớp học và lịch trình có thể được truy cập dễ dàng và nhanh chóng bởi cả sinh viên và nhân viên quản lý. Điều này giúp tối ưu hóa quá trình đăng ký và giảm thiểu sự cố giao thông thông tin.

Để tổ chức quản lý hệ thống đăng ký học phần theo cơ chế tín chỉ, trường đại học cần xây dựng một hệ thống thông tin mạnh mẽ và linh hoạt. Mỗi sinh viên sẽ được phân vào các khoa tương ứng với ngành học của mình để thực hiện việc đăng ký.

Trong mỗi kỳ học, việc lựa chọn môn học cho sinh viên đăng ký là một quá trình quan trọng được trường đại học tiến hành. Điều này bao gồm việc xác định các môn bắt buộc và môn tự chọn dựa trên chương trình học của từng ngành.

Mỗi môn học sẽ được đặc tả kỹ lưỡng với danh sách các môn tiên quyết cần thiết. Việc này đảm bảo rằng sinh viên chỉ đăng ký vào các môn học phù hợp với trình độ và kiến thức hiện tại của họ. Đồng thời, điều này cũng giúp sinh viên xác định được hướng nghiệp vụ và phát triển năng lực chuyên môn một cách mạnh mẽ.

Hệ thống quản lý sẽ thiết lập một ngưỡng tối đa là 30 tín chỉ cho mỗi kỳ học. Quy định này nhằm đảm bảo rằng sinh viên sẽ không quá tải về khối lượng học phí và có thể tập trung hơn vào việc hoàn thành các môn học một cách chất lượng.

Để ngăn chặn tình trạng đăng ký đồng thời hủy bỏ sau đó, sinh viên sẽ phải hoàn tất quá trình xác nhận trước khi họ có thể kết thúc quá trình đăng ký. Điều này giúp tăng cường sự chắc chắn và cam kết từ phía sinh viên đối với quyết định đăng ký môn học của họ.

Vào ngày mở đăng ký, sinh viên sẽ truy cập vào hệ thống thông qua giao diện trực tuyến. Tại đây, họ sẽ được hiển thị danh sách các môn học mà trường đã mở để đăng ký. Dựa trên nhu cầu học tập và quan tâm cá nhân, sinh viên sẽ có thể lựa chọn các môn học phù hợp và tiến hành quá trình đăng ký.

Trong trường hợp các lớp học đã đủ số lượng sinh viên đăng ký, sinh viên sẽ không thể đăng ký vào các lớp đó trực tiếp. Thay vào đó, họ sẽ được đưa vào danh sách dự bị. Nhà trường sẽ xem xét số lượng sinh viên trong danh sách này và quyết định về việc mở thêm lớp để đáp ứng nhu cầu học tập của sinh viên.

Sau khi đăng ký thành công, sinh viên sẽ nhận được một email xác nhận từ hệ thống, thông báo về việc đăng ký thành công và hướng dẫn về việc đóng học phí để hoàn tất quá trình đăng ký.

Hệ thống cũng cung cấp cho sinh viên khả năng xem thông tin học tập chi tiết và tổng quan về quá trình học tập của mình. Điều này bao gồm thông tin về số tín chỉ đã đạt, số môn đã học, điểm môn học, điểm trung bình tích lũy, và thời khóa biểu theo tuần.

Khi sinh viên hoàn thành đủ số tín chỉ theo từng ngành học, họ sẽ có quyền đăng ký xét tốt nghiệp. Nếu đáp ứng đủ các tiêu chuẩn, sinh viên sẽ được cấp bằng tốt nghiệp và thông tin này sẽ được công khai trên trang web của nhà trường. Thông tin cá nhân của cựu sinh viên sẽ được lưu giữ để theo dõi quá trình làm việc và các hoạt động sau khi tốt nghiệp, với sự đồng ý của sinh viên.

## Mục tiêu đề bài

Phát triển một hệ thống quản lý lớp học tín chỉ,, cung cấp khả năng đăng kí học tín chỉ và truy cập dễ dàng và sử dụng hiệu quả cho sinh viên.

## Mô tả yêu cầu chức năng

Quản lý người dùng (User Management):

* Đăng nhập/Đăng xuất: Sinh viên có thể đăng nhập vào hệ thống bằng email và mật khẩu đã đăng ký, và đăng xuất khi không sử dụng.
* Quản lý hồ sơ cá nhân: Sinh viên có thể cập nhật các thông tin cá nhân của mình.

Quản lý học phần (Course Management):

* Thêm/Sửa/Xóa học phần: Nhà trường có thể thêm, chỉnh sửa hoặc xóa các thông tin về học phần, bao gồm tên học phần, mã học phần, số tín chỉ, mô tả học phần, và các môn học tiên quyết.
* Quản lý các lớp học phần (Class Section Management): Nhà trường có thể tạo các lớp học phần với số lượng sinh viên tối đa, thời gian và địa điểm học.

Đăng ký học phần (Course Registration):

* Xem danh sách học phần có thể đăng ký: Sinh viên có thể xem danh sách các học phần mở trong kỳ học mà họ có đủ điều kiện để đăng ký.
* Đăng ký học phần: Sinh viên có thể chọn và đăng ký vào các lớp học phần mong muốn, hệ thống sẽ kiểm tra điều kiện tiên quyết và số lượng tín chỉ tối đa cho mỗi kỳ học.
* Xác nhận đăng ký: Sinh viên phải xác nhận đăng ký để tránh trường hợp đăng ký xong rồi hủy bỏ.

Quản lý thông tin học tập (Academic Information Management):

* Xem thông tin học tập cá nhân: Sinh viên có thể xem số tín chỉ đã đạt, số môn đã học, điểm môn học, điểm trung bình tích lũy, v.v.
* Xem thời khóa biểu: Sinh viên có thể xem thời khóa biểu theo tuần của mình

## Mô tả yêu cầu phi chức năng

Hệ thống đăng ký học phần theo cơ chế tín chỉ cần đáp ứng các yêu cầu phi chức năng nhằm đảm bảo hiệu suất, bảo mật, độ tin cậy, và khả năng sử dụng cao. Hệ thống phải đảm bảo thời gian phản hồi nhanh và khả năng xử lý đồng thời cao, cho phép mở rộng dễ dàng bằng cách thêm các dịch vụ mới mà không làm gián đoạn hoạt động. Về bảo mật, hệ thống phải bảo vệ dữ liệu nhạy cảm bằng cách mã hóa trong quá trình truyền và lưu trữ, đồng thời ngăn chặn các tấn công phổ biến. Khả năng bảo trì cũng rất quan trọng, với việc hỗ trợ cập nhật và nâng cấp dễ dàng, cùng với ghi log và giám sát chi tiết. Tất cả những yếu tố này đảm bảo hệ thống hoạt động hiệu quả, an toàn, và đáp ứng tốt nhu cầu của người dùng.

# Chương 2 PHÂN TÍCH VÀ THIẾT KẾ

## Danh sách tác nhân và mô tả

|  |  |  |
| --- | --- | --- |
| Tác nhân | Mô tả tác nhân | Ghi chú |
| Sinh viên | Là người sử dụng chính của hệ thống đăng ký học phần. Họ đăng nhập vào hệ thống để xem danh sách các môn học, đăng ký các môn học phù hợp, xác nhận đăng ký, và xem thông tin học tập cá nhân. |  |
| Giảng viên | là những người dạy các môn học và cần quản lý thông tin liên quan đến các môn học mà họ giảng dạy. |  |
| Admin | Là người quản lý ,điều hành và theo dõi mọi hoạt động của hệ thống đồng thời quản lý hệ thống đăng kí học tín ch |  |

## Sơ đồ class tổng quát

A diagram of a diagram

Description automatically generated with medium confidence

# Chương 3: KIẾN TRÚC

## Chọn kiến trúc:

Để thiết kế hệ thống quản lý lớp học tín chỉ cho một trường đại học như đã mô tả, kiến trúc phần mềm Microservices sẽ là lựa chọn tối ưu. Dưới đây là giải thích chi tiết và lý do lựa chọn kiến trúc này:

1. Lý do chọn kiến trúc

Để thiết kế hệ thống quản lý lớp học tín chỉ cho một trường đại học như đã mô tả, kiến trúc phần mềm Microservices sẽ là lựa chọn tối ưu. Dưới đây là giải thích chi tiết và lý do lựa chọn kiến trúc này:

1. Tính Modularity (Tính mô-đun hóa):

Microservices chia hệ thống thành các dịch vụ nhỏ gọn, mỗi dịch vụ đảm nhận một chức năng cụ thể. Trong trường hợp của hệ thống quản lý lớp học tín chỉ, các dịch vụ có thể bao gồm:

* User Service: Quản lý thông tin sinh viên, giảng viên, và các người dùng khác.
* Course Service: Quản lý thông tin các môn học, bao gồm môn học tiên quyết.
* Registration Service: Xử lý việc đăng ký và hủy đăng ký môn học.
* Notification Service: Gửi email xác nhận và thông báo.
* Schedule Service: Quản lý thời khóa biểu của sinh viên.
* Academic Records Service: Quản lý điểm số, tín chỉ đã đạt và thông tin học tập.
* Graduation Service: Xét duyệt tốt nghiệp và quản lý thông tin cựu sinh viên.

1. Scalability (Khả năng mở rộng):

Microservices cho phép mở rộng từng dịch vụ độc lập. Ví dụ, vào thời điểm đăng ký học phần, Registration Service có thể cần mở rộng nhiều hơn để đáp ứng số lượng lớn yêu cầu từ sinh viên. Các dịch vụ khác như User Service hoặc Notification Service có thể không cần mở rộng cùng lúc.

1. Resilience (Tính chịu lỗi):

Nếu một dịch vụ trong hệ thống gặp sự cố, nó không ảnh hưởng đến toàn bộ hệ thống. Ví dụ, nếu Notification Service gặp lỗi, hệ thống vẫn có thể cho phép sinh viên đăng ký học phần nhưng có thể trì hoãn việc gửi email thông báo.

1. Independent Deployment (Triển khai độc lập):

Các dịch vụ có thể được phát triển, triển khai, và bảo trì độc lập mà không ảnh hưởng đến các dịch vụ khác. Điều này giúp tăng tốc độ phát triển và triển khai các tính năng mới.

1. Technology Diversity (Đa dạng công nghệ):

Mỗi dịch vụ có thể được phát triển bằng các công nghệ phù hợp nhất với yêu cầu của nó. Ví dụ, một số dịch vụ có thể sử dụng Node.js cho tốc độ và khả năng xử lý I/O cao, trong khi các dịch vụ khác có thể sử dụng Java hoặc .NET cho các yêu cầu tính toán phức tạp.

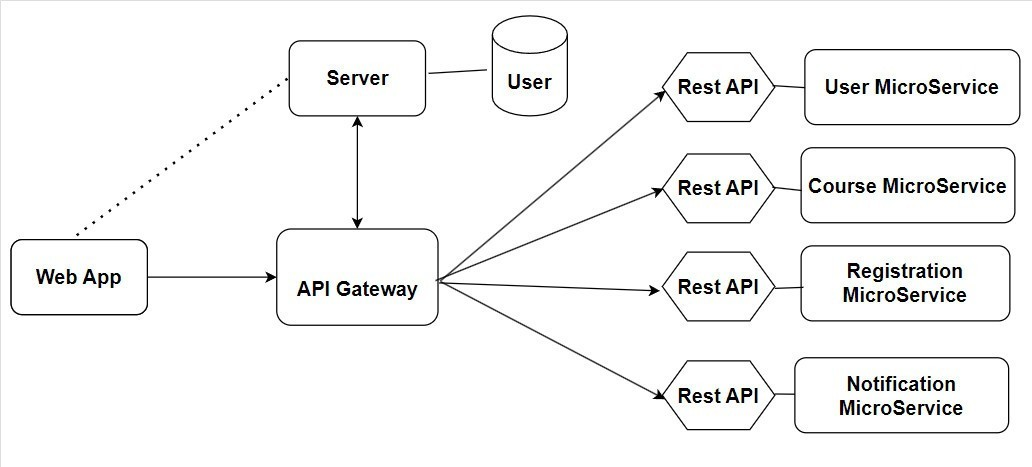
1. Data Management (Quản lý dữ liệu):

Mỗi microservice có thể có cơ sở dữ liệu riêng của nó, giúp giảm thiểu xung đột dữ liệu và tăng hiệu suất. Ví dụ, User Service có thể sử dụng một cơ sở dữ liệu quan hệ để quản lý thông tin người dùng, trong khi Schedule Service có thể sử dụng cơ sở dữ liệu NoSQL để lưu trữ thời khóa biểu.

Với kiến trúc Microservices, hệ thống quản lý lớp học tín chỉ sẽ có khả năng mở rộng linh hoạt, chịu lỗi tốt, và dễ dàng bảo trì và nâng cấp. Điều này sẽ giúp trường đại học đáp ứng tốt hơn các yêu cầu phức tạp và đa dạng từ sinh viên và các bộ phận liên quan.

## Thiết kế

Sơ đồ trình tự



Sơ đồ tổng quát (đầy đủ các thành phần):

A diagram of a web service

Description automatically generated

1. Cấu trúc code

A screenshot of a computer program

Description automatically generated

## So Sánh Với Những Kiến Trúc Khác

So sánh kiến trúc Microservices với một số kiến trúc khác, chúng ta có thể xem xét các khía cạnh chính sau đây: Monolithic Architecture, Service-Oriented Architecture (SOA), và Serverless Architecture. Mỗi kiến trúc có những ưu điểm và nhược điểm riêng tùy thuộc vào ngữ cảnh và yêu cầu cụ thể của ứng dụng.

Monolithic Architecture

Ưu điểm:

1. Đơn giản trong triển khai: Toàn bộ ứng dụng được đóng gói trong một gói triển khai duy nhất, giúp đơn giản hóa việc triển khai và quản lý.

2. Hiệu suất giao tiếp nội bộ: Không cần giao tiếp qua mạng giữa các thành phần của ứng dụng, giảm độ trễ và chi phí giao tiếp.

3. Dễ dàng phát triển ban đầu: Dễ bắt đầu phát triển với một cơ sở mã duy nhất và dễ dàng quản lý trong giai đoạn đầu.

Nhược điểm:

1. Khó mở rộng: Mở rộng một phần của ứng dụng đòi hỏi phải mở rộng toàn bộ ứng dụng, dẫn đến lãng phí tài nguyên.

2. Khó bảo trì và phát triển: Khi ứng dụng lớn lên, việc bảo trì và thêm tính năng mới trở nên khó khăn hơn.

3. Khả năng chịu lỗi kém: Nếu một phần của ứng dụng gặp sự cố, toàn bộ ứng dụng có thể bị ảnh hưởng.

Service-Oriented Architecture (SOA)

Ưu điểm:

1. Tái sử dụng dịch vụ: Các dịch vụ có thể được tái sử dụng bởi nhiều ứng dụng khác nhau.

2. Phân tách rõ ràng: Các dịch vụ có sự phân tách rõ ràng, dễ quản lý và phát triển độc lập.

Nhược điểm:

1. Phức tạp trong triển khai: SOA thường sử dụng các tiêu chuẩn phức tạp như SOAP và WS dẫn đến khó khăn trong triển khai và tích hợp.

2. Hiệu suất giao tiếp thấp hơn: Giao tiếp giữa các dịch vụ có thể bị chậm do sử dụng các giao thức phức tạp.

3. Quản lý giao tiếp khó khăn: Quản lý giao tiếp giữa các dịch vụ có thể trở nên phức tạp và khó khăn.

Serverless Architecture

Ưu điểm:

1. Không cần quản lý hạ tầng: Hạ tầng được quản lý bởi nhà cung cấp dịch vụ, giúp giảm bớt gánh nặng quản lý.

2. Tự động mở rộng: Các chức năng có thể tự động mở rộng dựa trên nhu cầu thực tế, tối ưu hóa tài nguyên.

3. Thanh toán theo sử dụng: Chỉ trả tiền cho tài nguyên đã sử dụng, giúp giảm chi phí.

Nhược điểm:

1. Độ trễ lạnh: Chức năng có thể bị trễ khi khởi động lần đầu do cần phải "hâm nóng".

2. Giới hạn của nhà cung cấp: Bị giới hạn bởi các quy định và giới hạn của nhà cung cấp dịch vụ serverless.

3. Khó khăn trong phát triển và gỡ lỗi: Khó phát triển và gỡ lỗi do môi trường chạy không liên tục.

**So sánh với Microservices Architecture**

Ưu điểm của Microservices:

1. Khả năng mở rộng linh hoạt: Từng dịch vụ có thể được mở rộng độc lập, tối ưu hóa tài nguyên.

2. Tính chịu lỗi cao: Sự cố trong một dịch vụ không ảnh hưởng đến toàn bộ hệ thống.

3. Phát triển và triển khai độc lập: Các đội ngũ có thể phát triển, triển khai và quản lý các dịch vụ độc lập với nhau.

4. Đa dạng công nghệ: Có thể sử dụng các công nghệ khác nhau cho từng dịch vụ dựa trên yêu cầu cụ thể.

Nhược điểm của Microservices:

1. Phức tạp trong triển khai và quản lý: Quản lý nhiều dịch vụ độc lập có thể trở nên phức tạp, đòi hỏi các công cụ quản lý và triển khai chuyên dụng.

2. Giao tiếp qua mạng: Giao tiếp giữa các dịch vụ qua mạng có thể dẫn đến độ trễ và tăng chi phí giao tiếp.

3. Khó gỡ lỗi và giám sát: Việc gỡ lỗi và giám sát trở nên phức tạp hơn do sự phân tán của các dịch vụ.

Microservices Architecture là lựa chọn tốt cho các hệ thống lớn, phức tạp và yêu cầu khả năng mở rộng cao, mặc dù việc triển khai và quản lý có thể phức tạp hơn so với Monolithic Architecture. Nó cung cấp sự linh hoạt, chịu lỗi và khả năng mở rộng tốt hơn nhưng yêu cầu đầu tư vào công cụ quản lý và kiến thức kỹ thuật cao hơn.

# Chương 4: TỔNG KẾT

Nhóm đã thiết kế và triển khai một hệ thống đăng ký học phần sử dụng kiến trúc microservice. Đây là một phương pháp phổ biến để phân chia ứng dụng thành các dịch vụ nhỏ, độc lập và có thể mở rộng.

## Tổng quan kiến trúc microservice:

Kiến trúc microservice chia ứng dụng thành nhiều dịch vụ nhỏ, mỗi dịch vụ đảm nhận một chức năng cụ thể và giao tiếp với nhau thông qua các giao diện lập trình ứng dụng (API). Cấu trúc này giúp tách biệt các thành phần của hệ thống, làm cho việc quản lý, bảo trì và mở rộng dễ dàng hơn.

## Kết quả đạt được

Tính độc lập và phân tách rõ ràng: Các dịch vụ được phân tách rõ ràng theo chức năng, giúp việc phát triển và bảo trì dễ dàng hơn.

Khả năng mở rộng: Hệ thống có thể dễ dàng mở rộng bằng cách thêm các dịch vụ mới hoặc mở rộng các dịch vụ hiện có mà không ảnh hưởng đến toàn bộ hệ thống. Việc triển khai các dịch vụ mới hoặc nâng cấp các dịch vụ hiện tại có thể được thực hiện mà không cần dừng toàn bộ hệ thống, giúp duy trì tính liên tục trong hoạt động.

Cải thiện thời gian phát triển và triển khai: Do các dịch vụ được phân tách độc lập, quá trình phát triển và triển khai có thể được thực hiện song song. Điều này rút ngắn thời gian phát triển tổng thể và cho phép đưa các tính năng mới ra thị trường nhanh hơn.

Dễ dàng trong việc kiểm tra và bảo trì: Việc phân chia thành các dịch vụ nhỏ giúp kiểm tra và bảo trì dễ dàng hơn. Mỗi dịch vụ có thể được kiểm tra độc lập, và các vấn đề có thể được phát hiện và khắc phục nhanh chóng mà không ảnh hưởng đến các phần khác của hệ thống.

## Những Hạn Chế

Trong quá trình thực hiện đồ án về Kiến Trúc và Thiết Kế Phần Mềm, chúng em đã gặp phải một số hạn chế và khó khăn sau đây:

1. Thiếu Kinh Nghiệm Thực Tế: Do thiếu kinh nghiệm thực tế, chúng em gặp khó khăn trong việc áp dụng các nguyên lý và mô hình kiến trúc một cách hiệu quả. Các quyết định thiết kế có thể chưa tối ưu và chưa tuân theo các tiêu chuẩn công nghiệp hiện đại.

2. Kiến Thức Lý Thuyết Hạn Chế: Mặc dù đã học qua các khái niệm cơ bản về kiến trúc phần mềm, nhưng kiến thức lý thuyết của chúng em vẫn còn hạn chế. Điều này dẫn đến khó khăn trong việc lựa chọn và áp dụng các mẫu kiến trúc phù hợp cho bài toán cụ thể. Chúng em nhận thấy cần phải bổ sung thêm kiến thức chuyên sâu và cập nhật các xu hướng mới trong ngành.

3. Chưa Hiện Thực Được Mã Nguồn:Một trong những mục tiêu của đồ án là hiện thực hóa mã nguồn để kiểm chứng các thiết kế và mô hình kiến trúc. Tuy nhiên, do hạn chế về thời gian và năng lực kỹ thuật, chúng em chưa thể hoàn thành việc hiện thực này. Điều này làm giảm khả năng đánh giá và kiểm chứng tính hiệu quả của các thiết kế đề xuất.

4. Hạn Chế Trong Việc Sử Dụng Công Cụ và Công Nghệ: Việc sử dụng các công cụ và công nghệ hỗ trợ thiết kế và phát triển phần mềm chưa được thực hiện một cách thành thạo. Các công cụ như UML, các phần mềm mô phỏng và quản lý dự án chưa được khai thác hết tiềm năng, dẫn đến việc trình bày và quản lý dự án còn nhiều hạn chế.

5. Phạm Vi Đề Tài Chưa Được Mở Rộng: Đồ án mới chỉ tập trung vào một số khía cạnh cơ bản của kiến trúc phần mềm mà chưa mở rộng ra các vấn đề phức tạp hơn như hiệu năng, bảo mật và khả năng mở rộng. Việc này làm giảm đi tính thực tiễn và toàn diện của đồ án

6. Thiếu Đánh Giá và Phản Hồi Từ Thực Tế: Trong quá trình thực hiện đồ án, chúng em chưa có điều kiện để nhận phản hồi từ các chuyên gia trong ngành hoặc từ các dự án thực tế. Điều này làm hạn chế khả năng cải thiện và tối ưu hóa thiết kế của chúng em.

## Hướng phát triển trong tương lai:

Mở rộng tính năng: Phát triển thêm các tính năng mới dựa trên phản hồi từ người dùng và yêu cầu của nhà trường. Tích hợp thêm các dịch vụ hỗ trợ như dịch vụ phân tích dữ liệu, dịch vụ hỗ trợ học tập trực tuyến, v.v.

Đảm bảo an ninh và bảo mật: Tăng cường các biện pháp bảo mật để bảo vệ dữ liệu người dùng và đảm bảo an toàn thông tin trong hệ thống. Thực hiện kiểm tra bảo mật thường xuyên và cập nhật các biện pháp bảo vệ mới nhất.

Nâng cao trải nghiệm người dùng: Tiếp tục cải thiện giao diện và trải nghiệm người dùng dựa trên phản hồi và phân tích hành vi người dùng.

Tóm lại, việc áp dụng kiến trúc microservice cho hệ thống đăng ký học phần đã mang lại nhiều lợi ích về tính linh hoạt, khả năng mở rộng và quản lý. Đây là một hướng tiếp cận hiệu quả cho việc phát triển các hệ thống lớn và phức tạp trong môi trường công nghệ hiện đại.