**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**



**quản lý Đăng kí học phần**

***Nhóm 03 - Sinh viên thực hiện***

1. Lê Xuân Tuấn Anh - 20074191
2. Phạm Ngọc Huy Hoàng – 20091411
3. Phạm Thanh Nhật – 20091481

# MỤC LỤC

[CHƯƠNG 1 : GIỚI THIỆU 2](#_Toc167123174)

[1.1 Tổng quan 2](#_Toc167123175)

[1.2 Mục tiêu đề tài 2](#_Toc167123176)

[1.3 Phạm vi đề tài 2](#_Toc167123177)

[1.3.1.2 Chức năng dành cho người quản trị 3](#_Toc167123178)

[1.4 Mô tả yêu cầu chức năng 3](#_Toc167123179)

[1.4.1.2 Mô tả hoạt động của hệ thống 3](#_Toc167123180)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 5](#_Toc167123181)

[2.1 Node.js 5](#_Toc167123182)

[2.2 MongoDB 6](#_Toc167123183)

[2.3 Express JS 7](#_Toc167123184)

[2.4 JavaScript 9](#_Toc167123185)

[2.5 Kiến trúc 10](#_Toc167123186)

[CHƯƠNG 3 : PHÂN TÍCH VÀ THIẾT KẾ 13](#_Toc167123187)

[3.1 Phân tích hệ thống 13](#_Toc167123188)

[3.1.1 Usecase tổng quát 14](#_Toc167123189)

[3.1.2 Danh sách các tình huống hoạt động (Use cases) 14](#_Toc167123190)

[3.1.3 Tình huống hoạt động 15](#_Toc167123191)

[3.2 Class diagram 20](#_Toc167123192)

[CHƯƠNG 4 : HIỆN THỰC 21](#_Toc167123193)

[4.1 Giao diện của hệ thống 21](#_Toc167123194)

[4.1.1 Giao diện web 21](#_Toc167123195)

[CHƯƠNG 5 : KẾT LUẬN 28](#_Toc167123196)

[5.1 Kết quả đạt được 28](#_Toc167123197)

[5.2 Hạn chế của đồ án 28](#_Toc167123198)

[5.3 Hướng phát triển 28](#_Toc167123199)

[TÀI LIỆU THAM KHẢO 29](#_Toc167123200)

# : GIỚI THIỆU

## Tổng quan

Hệ thống quản lý lớp học tín chỉ của trường đại học là một công cụ điện tử cần thiết, giúp tổ chức và quản lý các hoạt động giảng dạy và học tập một cách hiệu quả. Hệ thống này cho phép sinh viên đăng ký các môn học, theo dõi tiến trình học tập, và xem điểm số qua các học kỳ. Đồng thời, hệ thống cũng cung cấp cho giảng viên khả năng quản lý danh sách lớp, đánh giá và ghi chép điểm số sinh viên một cách dễ dàng và chính xác. Với sự hỗ trợ của công nghệ thông tin, hệ thống này giảm bớt gánh nặng hành chính cho phòng đào tạo và giảng viên, đồng thời tăng cường tính minh bạch và truy cập thông tin trong môi trường giáo dục. Hệ thống quản lý lớp học tín chỉ cũng giúp các trường đại học có thể phân tích dữ liệu học tập, từ đó đưa ra các quyết định chiến lược để nâng cao chất lượng giáo dục và đáp ứng tốt hơn nhu cầu của sinh viên.

## Mục tiêu đề tài

Hệ thống quản lý lớp học tín chỉ của trường đại học mang lại khả năng tương tác cao và tiện lợi cho sinh viên, thông qua nhiều tính năng hữu ích. Sinh viên có thể truy cập và chỉnh sửa thông tin cá nhân của mình, xem lịch học và kết quả học tập một cách dễ dàng. Hệ thống cũng cho phép sinh viên tra cứu công nợ, theo dõi lịch học theo tiến độ và tiến trình học tập cá nhân. Ngoài ra, sinh viên có thể đăng ký các học phần mong muốn. Tất cả các tính năng này được tích hợp trên một nền tảng website, giúp sinh viên quản lý học tập của mình một cách hiệu quả và thuận tiện, từ đó nâng cao trải nghiệm giáo dục tổng thể.

## Phạm vi đề tài

#### Chức năng dành cho người dùng:

* Sinh viên có thể xem thông tin cá nhân của mình, đăng ký học phần mong muốn, xem lịch học, xem lịch trùng, xem kết quả học tập, tra cứu công nợ, xem lịch học theo tiến độ.
* Có giao diện thân thiện, dễ sử dụng, để có thể tương tác một cách thuận tiện nhất.
* Có thể thay đổi : thông tin tài khoản...

#### **Chức năng dành cho người quản trị**:

Ngoài các yêu cầu giống như của người dùng, thì quản trị viên phải đảm bảo những yêu cầu sau:

* Thêm, xóa, chỉnh sửa quản lý học phần, khoa, chuyên ngành, sinh viên, môn học, lớp học phần, giảng viên, học kì

## Mô tả yêu cầu chức năng

#### Phân tích yêu cầu của hệ thống

- Người dùng có thể thay đổi thông tin tài khoản như: mật khẩu...

- Người dùng đăng ký học phần mong muốn trên web, hệ thống sẽ tiếp nhận và phản hồi cho đối tượng đăng kí học phần.

- Người dùng có thể hủy đăng ký học phần.

- Từ mô tả trên, có thể đưa ra yêu cầu của hệ thống với hai đối tượng chính tương tác với hệ thống như sau:

• Đối với quản trị viên:

- Thêm, xóa, chỉnh sửa quản lý học phần, khoa, chuyên ngành, sinh viên, môn học, lớp học phần, giảng viên, học kì

• Đối với sinh viên:

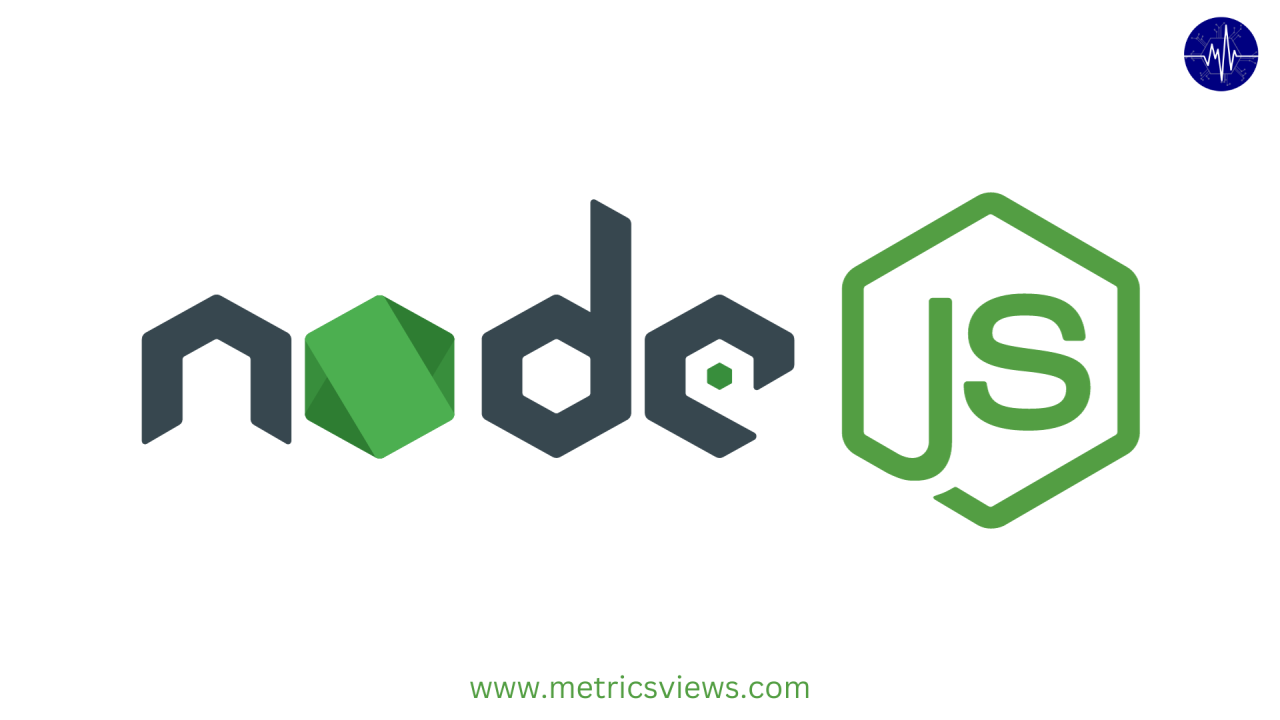
- Sinh viên có thể xem thông tin cá nhân của mình, đăng ký học phần mong muốn, xem lịch học, xem lịch trùng, tra cứu công nợ, và hủy đăng ký học phần.

#### Mô tả hoạt động của hệ thống

1. **Phân loại sinh viên theo ngành**: Dựa vào thông tin đăng ký ban đầu, hệ thống tự động phân bổ sinh viên vào các ngành học và khoa tương ứng, đảm bảo mỗi sinh viên được xếp vào lớp học phù hợp với ngành học của họ.
2. **Lựa chọn môn học mở trong học kỳ:** Các khoa sẽ chọn lọc và nhập danh sách môn học cần mở cho mỗi ngành vào hệ thống, tạo điều kiện cho sinh viên lựa chọn.
3. **Tiến trình đăng ký học phần:** Sinh viên đăng nhập hệ thống để xem các môn học được mở và tiến hành đăng ký những môn họ muốn học trong học kỳ.
4. **Xác nhận đăng ký**: Trước khi hệ thống chính thức chấp nhận, sinh viên cần xác nhận đăng ký của mình để tránh các sai sót như đăng ký nhầm hoặc cần thay đổi sau này.
5. **Quản lý đăng ký**: Sau khi đăng ký, hệ thống sẽ kiểm tra và xử lý thông tin, bao gồm số tín chỉ và số lượng sinh viên đăng ký mỗi lớp, để đảm bảo không quá tải.
6. **Theo dõi thông tin học tập**: Sinh viên có thể kiểm tra các thông tin như số tín chỉ đã tích lũy, các môn đã hoàn thành, điểm số từng môn và điểm trung bình tích lũy qua hệ thống.
7. **Kiểm tra lịch học cá nhân:** Hệ thống cung cấp lịch học chi tiết cho mỗi sinh viên, bao gồm thời gian, địa điểm và các chi tiết khác của từng lớp học, giúp sinh viên sắp xếp thời gian học tập hiệu quả.Top of Form

# : CƠ SỞ LÝ THUYẾTTop of Form

## Node.js



Hình 2‑1 Node.js development

**Tổng quan về Node.js:**

Node.js là một nền tảng phát triển đa năng, dựa trên JavaScript V8 Engine của Google Chrome, tạo điều kiện thuận lợi cho việc phát triển các ứng dụng mạng hiệu quả và dễ dàng mở rộng

**Ưu điểm của Node.js:**

1. **Đồng bộ hóa ngôn ngữ:** Node.js sử dụng JavaScript, ngôn ngữ lập trình được ưa chuộng rộng rãi, cho phép các nhà phát triển sử dụng cùng một ngôn ngữ cho cả phía client và server, giúp đơn giản hóa quá trình phát triển.
2. **Kiến trúc hướng sự kiện:** Node.js được thiết kế với kiến trúc hướng sự kiện và non-blocking I/O, điều này không chỉ giúp tăng hiệu suất mà còn cải thiện khả năng xử lý các yêu cầu đồng thời, làm cho nó trở thành giải pháp tối ưu cho các ứng dụng đòi hỏi khả năng xử lý kết nối cao.
3. **Tính linh hoạt cao:** Node.js hỗ trợ nhiều hệ điều hành khác nhau bao gồm macOS, Windows và Linux, mang lại sự linh hoạt và thuận tiện trong triển khai trên nhiều môi trường phát triển khác nhau.
4. **Cộng đồng lớn:** Node.js có một cộng đồng người dùng và nhà phát triển rất lớn và năng động, cùng với thư viện mô-đun npm phong phú, hỗ trợ nhanh chóng và hiệu quả, từ đó giúp tăng tốc quá trình phát triển và đối phó với các thách thức kỹ thuật.
5. **Mã nguồn mở và miễn phí:** Là một nền tảng mã nguồn mở, Node.js cung cấp miễn phí cho cộng đồng nhà phát triển, giúp giảm thiểu chi phí phát triển ứng dụng.

Với những đặc điểm này, Node.js đã và đang trở thành một sự lựa chọn phổ biến và được ưa chuộng trong cộng đồng phát triển web và ứng dụng di động trên toàn cầu.Top of Form

Top of Form

## MongoDB



**Tổng quan về MongoDB:**

MongoDB là một hệ thống quản lý cơ sở dữ liệu không quan hệ (NoSQL), phát triển dựa trên nguyên tắc mã nguồn mở. Nó được thiết kế để cung cấp một cơ chế lưu trữ dữ liệu mềm dẻo, không yêu cầu một cấu trúc cố định giống như các hệ thống quản lý cơ sở dữ liệu quan hệ truyền thống.

**Ưu điểm của MongoDB:**

1. **Tối ưu hiệu suất:** MongoDB được thiết kế để cung cấp hiệu suất truy xuất và xử lý dữ liệu nhanh chóng, đáp ứng nhu cầu của các ứng dụng hiện đại.
2. **Độ tin cậy cao:** Hỗ trợ nhân rộng dữ liệu một cách tự động, MongoDB giúp đảm bảo tính sẵn sàng và độ bền của dữ liệu, phù hợp với các hệ thống cần tính chịu lỗi cao.
3. **Khả năng mở rộng:** Với tính năng sharding, MongoDB có khả năng mở rộng ngang để xử lý lượng lớn dữ liệu, chia nhỏ dữ liệu thành các phân đoạn để quản lý và xử lý hiệu quả hơn.
4. **Cấu trúc linh hoạt:** Không yêu cầu sơ đồ cố định, MongoDB cho phép các nhà phát triển thêm hoặc loại bỏ trường dữ liệu mà không làm ảnh hưởng đến các ứng dụng đang chạy.
5. **Lưu trữ và truy vấn linh hoạt:** Hỗ trợ lưu trữ dữ liệu theo định dạng JSON/BSON, cho phép thực hiện các truy vấn dữ liệu phức tạp và đa dạng.
6. **Dữ liệu phi cấu trúc:** Cho phép lưu trữ các bản ghi không đồng nhất trong cùng một bộ sưu tập, điều này là lý tưởng cho việc xử lý dữ liệu không cố định cấu trúc.
7. **Giảm sử dụng joins:** MongoDB tránh sử dụng joins bằng cách áp dụng mô hình nhúng dữ liệu, giúp cải thiện đáng kể hiệu suất truy vấn.
8. **Phân phối đa nền tảng:** Có thể triển khai trên nhiều hệ điều hành và môi trường, từ máy chủ cá nhân đến hệ thống đám mây phức tạp.

Những đặc điểm này giúp MongoDB trở thành lựa chọn ưu tiên cho các ứng dụng cần xử lý lượng lớn dữ liệu động và có yêu cầu cao về khả năng mở rộng và tính linh hoạt.

## Express JS



**Tổng quan về ExpressJS:**

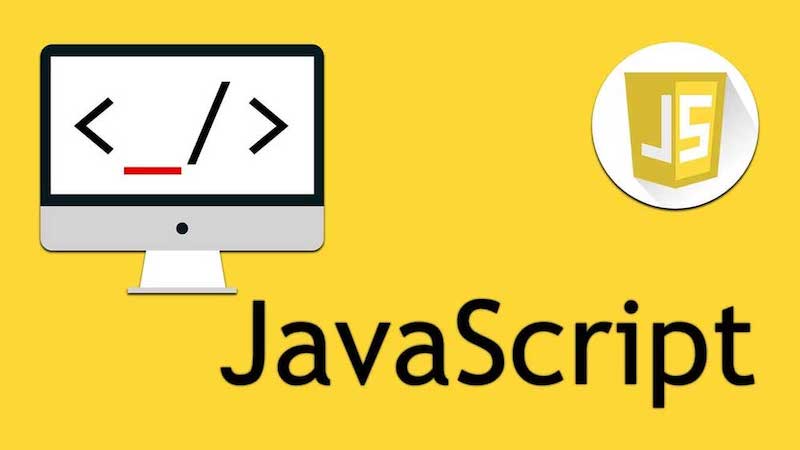
MongoDB là một hệ thống quản lý cơ sở dữ liệu không quan hệ (NoSQL), phát triển dựa trên nguyên tắc mã nguồn mở. Nó được thiết kế để cung cấp một cơ chế lưu trữ dữ liệu mềm dẻo, không yêu cầu một cấu trúc cố định giống như các hệ thống quản lý cơ sở dữ liệu quan hệ truyền thống.

**Ưu điểm của ExpressJS:**

1. **Cấu trúc đơn giản và gọn nhẹ:** Express.js cung cấp một cách tiếp cận tối giản nhưng mạnh mẽ để xây dựng các ứng dụng web, giúp giảm thiểu độ phức tạp trong quản lý tài nguyên và xử lý các yêu cầu.
2. **Tích hợp mạnh mẽ:** Express.js làm việc rất tốt với các công nghệ khác trong hệ sinh thái Node.js, đặc biệt là với MongoDB và AngularJS, tạo thành mô hình MEAN stack, cho phép phát triển full-stack JavaScript hiệu quả.
3. **Quản lý tuyến đường dễ dàng:** Express.js cung cấp các công cụ rất linh hoạt để xử lý tuyến đường (routing), cho phép các nhà phát triển tạo ra các ứng dụng với nhiều trang và các RESTful API dễ dàng.
4. **Middleware:** Khả năng sử dụng middleware cho phép các nhà phát triển thêm các chức năng xử lý trước và sau khi yêu cầu HTTP được xử lý, điều này làm cho Express.js trở thành công cụ lý tưởng để xây dựng các API phức tạp.
5. **Hiệu suất cao:** Express.js tối ưu hóa các yêu cầu HTTP, giúp xây dựng các ứng dụng web và API có hiệu suất cao.
6. **Cộng đồng mạnh mẽ và hỗ trợ rộng rãi:** Là một phần của hệ sinh thái Node.js, Express.js được hỗ trợ bởi một cộng đồng lớn các nhà phát triển, cung cấp một lượng lớn tài nguyên học tập, plugin và công cụ phát triển.
7. **Phát triển nhanh:** Nhờ vào các tính năng tự động hóa và các công cụ sẵn có, Express.js cho phép phát triển nhanh chóng các ứng dụng web và API, từ đó rút ngắn thời gian đưa sản phẩm ra thị trường.

Nhờ vào những ưu điểm này, Express.js đã trở thành một trong những framework được ưa chuộng nhất cho việc phát triển ứng dụng web và API trong môi trường JavaScript.

## JavaScript



**Tổng quan về JavaScript:**

JavaScript là một ngôn ngữ lập trình đa năng, chủ yếu được sử dụng để thêm các tính năng tương tác và động vào các trang web. Đây là một trong ba ngôn ngữ cốt lõi của World Wide Web, cùng với HTML và CSS. Dưới đây là một số ưu điểm nổi bật của JavaScript

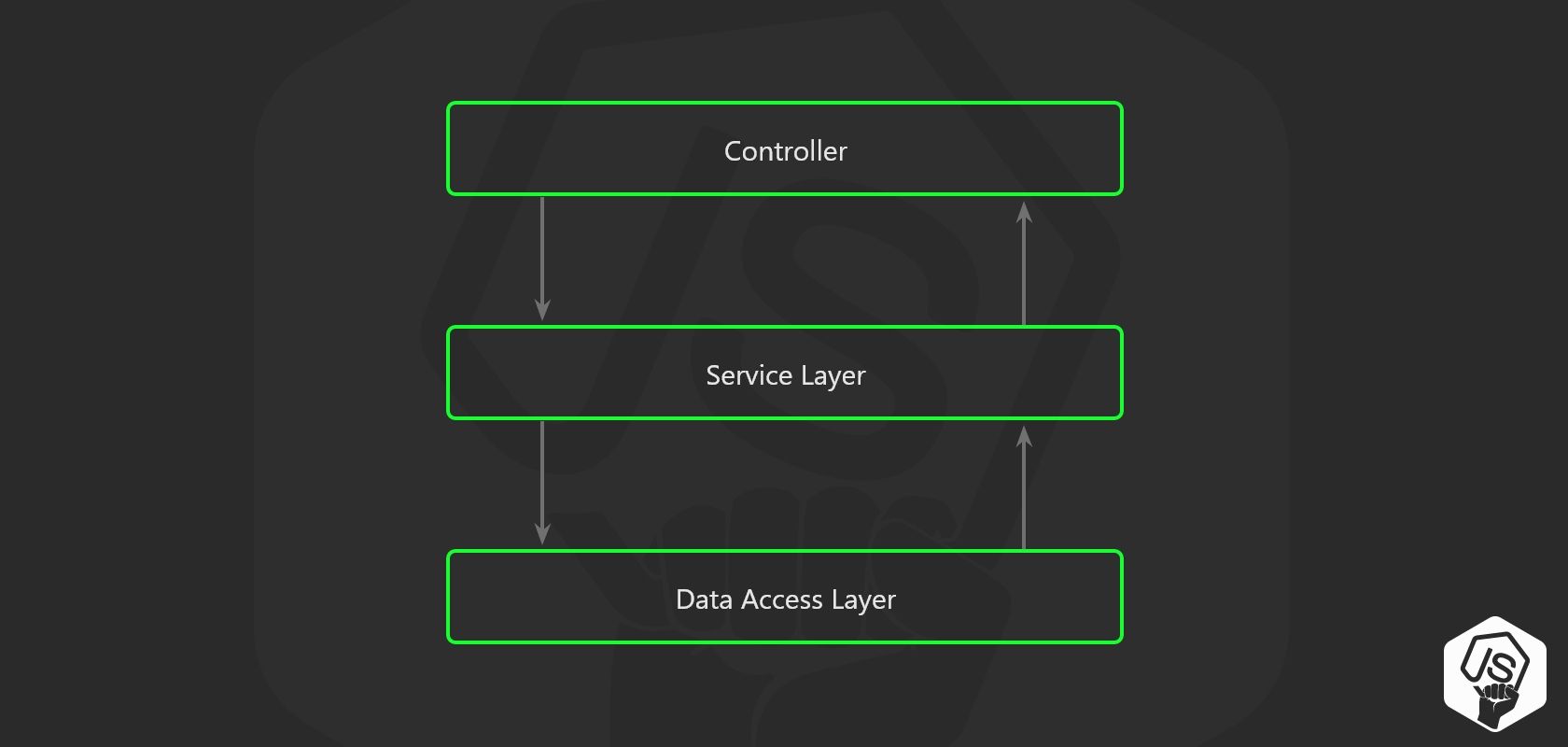
**Ưu điểm của JavaScript:**

1. **Đa nền tảng:** JavaScript được hỗ trợ trên hầu hết các trình duyệt web hiện đại, cho phép phát triển các ứng dụng có thể chạy mượt mà trên nhiều thiết bị và hệ điều hành khác nhau mà không cần thay đổi mã nguồn.
2. **Tính linh hoạt:** JavaScript không chỉ được sử dụng để phát triển ứng dụng web mà còn được sử dụng trong phát triển server (Node.js), ứng dụng di động (React Native), và thậm chí là trong lập trình robot và phát triển trò chơi.
3. **Hỗ trợ lập trình hướng đối tượng và hàm:** JavaScript hỗ trợ cả lập trình hướng đối tượng và hàm, cho phép các nhà phát triển tận dụng các kỹ thuật phát triển phần mềm tiên tiến để tạo ra các ứng dụng hiệu quả và dễ bảo trì.
4. **Thư viện và framework phong phú:** Có một số lượng lớn thư viện và framework dựa trên JavaScript, như React, Angular, và Vue.js, giúp tăng tốc quá trình phát triển và cung cấp các giải pháp mạnh mẽ cho các nhu cầu phức tạp.
5. **Cộng đồng mạnh mẽ:** JavaScript có một trong những cộng đồng lớn nhất thế giới, với hàng triệu nhà phát triển và một loạt các tài nguyên học tập, từ các khóa học trực tuyến, sách, blog, đến các hội nghị chuyên ngành.
6. **Phát triển nhanh và hiệu suất cao:** JavaScript cho phép phát triển nhanh chóng các ứng dụng web, và các cải tiến liên tục trong các trình duyệt đã làm tăng đáng kể hiệu suất thực thi của JavaScript.

Những ưu điểm này giúp JavaScript trở thành một trong những ngôn ngữ lập trình phổ biến nhất và là lựa chọn hàng đầu cho phát triển web và nhiều lĩnh vực công nghệ khác.

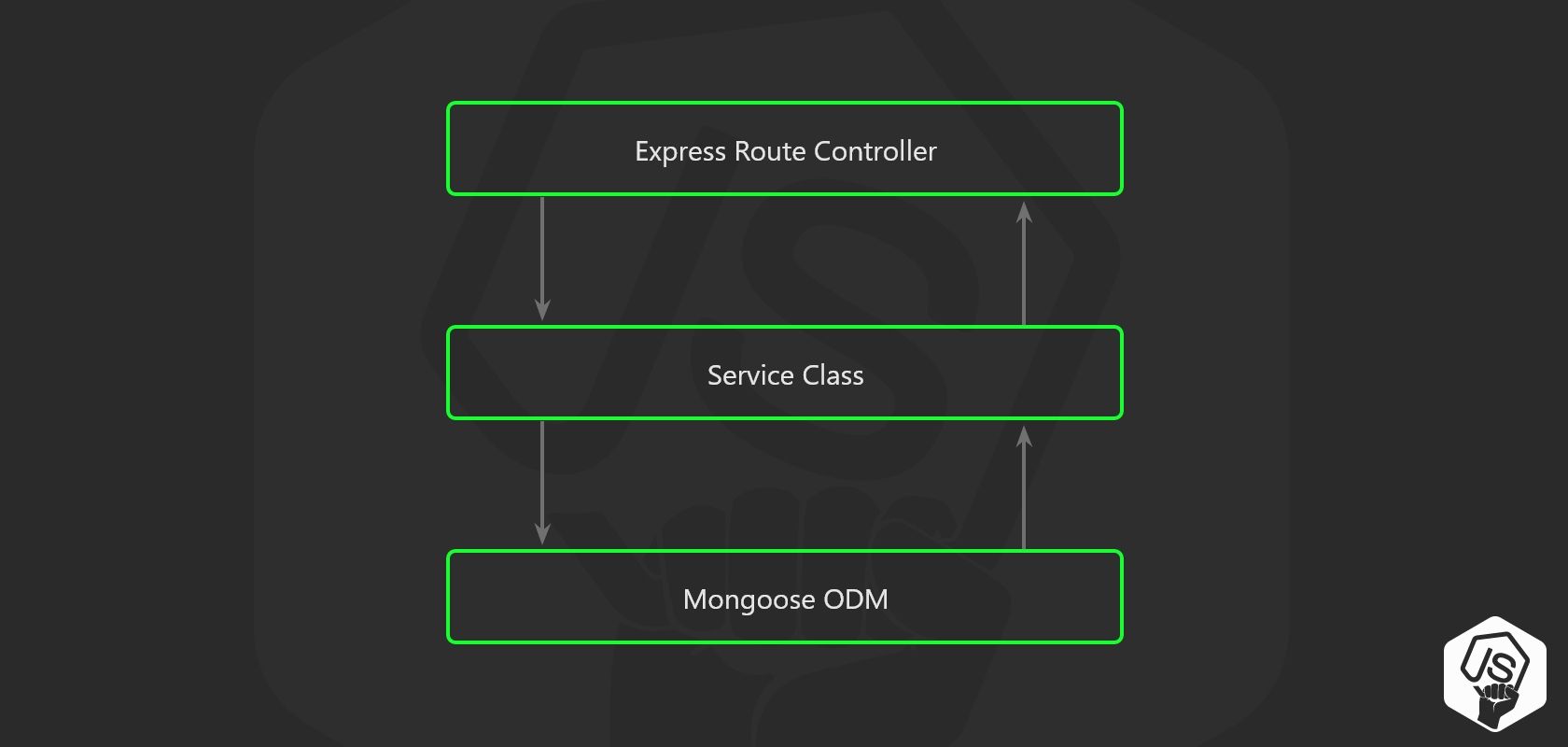
## Kiến trúc

Đề tài áp dụng kiến trúc 3 layer



Ý tưởng kiến trúc này là tách biệt hoàn toàn logic nghiệp vụ cũng như truy vấn database ra khỏi API route và chuyển chúng đến các layer tương ứng như bên trên.

Dưới đây là mô hình sau khi apply vào Express trong NodeJS:



1. Controller layer (API routes and endpoints) Đây là module code cho phép định nghĩa các API route trong ứng dụng. Nhiệm vụ module này là xác thức data và chuyển đến Service Layer để xử lý.
2. Service layer (for business logic) Module này cho phép tạo ra lớp tách biệt ra khỏi các API Router. Mỗi một service tương ứng với một nhiệm vụ riêng biệt (Nên tuân theo nguyên tắc SOLID). Việc chuyển logic nghiệp vụ vào module này cho phép các thể tái sử dụng ở các API khác nhau một cách hiệu quả.
3. Data access layer (for working with a database) Module này đảm nhận trách nhiệm giao tiếp với database với các công việc truy vấn như get, update, create,.... Ngoài ra các công việc như định nghĩa model, ORM, connect databse đều được thực hiện tại đây.
4. **Ưu điểm:**
   1. **Tính Modularity (Tính mô-đun)**: Kiến trúc layer cho phép hệ thống được phân chia thành các phần nhỏ, gọi là lớp, mỗi lớp đảm nhận một chức năng riêng biệt. Điều này làm cho hệ thống dễ quản lý và dễ mở rộng.
   2. **Tính Tương tác Thấp (Low Coupling)**: Các lớp không phụ thuộc trực tiếp lẫn nhau, mà thay vào đó chúng chỉ tương tác thông qua giao diện chuẩn hóa. Điều này giúp giảm sự phụ thuộc giữa các thành phần của hệ thống, làm cho việc thay đổi hoặc nâng cấp dễ dàng hơn.
   3. **Tính Đa dạng (Abstraction)**: Mỗi lớp ẩn đi chi tiết triển khai của các lớp dưới cùng, chỉ cung cấp một giao diện cho các lớp trên cùng. Điều này giúp tạo ra một cách nhìn trừu tượng và dễ hiểu về cách hoạt động của hệ thống.
5. **Nhược điểm:**
6. **Tăng Overhead (Tăng Chi phí)**: Mỗi lớp thêm vào đều tăng overhead cho hệ thống, từ việc tạo và duy trì các giao diện đến việc truyền thông qua các lớp. Điều này có thể ảnh hưởng đến hiệu suất của hệ thống, đặc biệt là trong các hệ thống có số lượng lớn các lớp.
7. **Khó Khắc phục Sự cố (Troubleshooting Difficulty)**: Khi một sự cố xảy ra, việc xác định lớp nào gây ra sự cố và nơi nó xảy ra có thể trở nên phức tạp. Điều này đặc biệt đúng khi mối quan hệ giữa các lớp phức tạp.
8. **Một số Khía cạnh Khác:**
9. **Tính Bảo mật (Security)**: Kiến trúc layer có thể được sử dụng để cải thiện bảo mật bằng cách tách biệt các lớp và áp dụng các biện pháp bảo mật tại mỗi lớp.
10. **Tính Tái sử dụng (Reusability)**: Các lớp có thể được thiết kế để có thể tái sử dụng, giúp giảm thiểu việc viết lại mã và tăng tốc độ phát triển phần mềm.

# : PHÂN TÍCH VÀ THIẾT KẾ

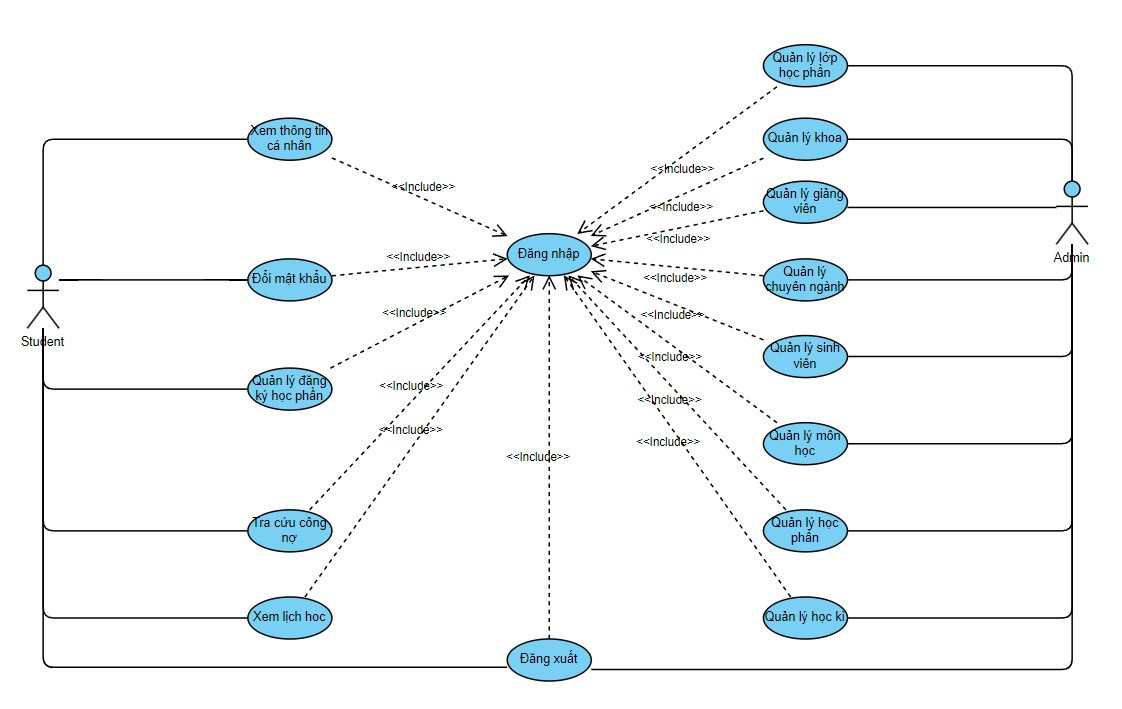
## Phân tích hệ thống

Hệ thống ASC-IUH được sử dụng bởi sinh viên và quản trị viên của trường. Trong đó, ASC-IUH tập trung vào việc phát triển các chức năng quan trọng của hệ thống mà nhà trường hiện đang gặp phải các vấn đề như đăng ký học phần. Ngoài ra, các chức năng của quản trị viên cũng được phát triển để hỗ trợ hệ thống đồng bộ các thao tác của sinh viên và các nghiệp vụ đào tạo.

|  |  |  |
| --- | --- | --- |
| **Đối Tượng** | **Tên Chức Năng** | **Mô Tả** |
| **Sinh viên** | Đăng nhập tài khoản | Đăng nhập vào hệ thống với mã số sinh viên được tạo |
| Xem thông tin cá nhân | Xem thông tin cá nhân của sinh viên đã đăng nhập vào ứng dụng |
| Quản lý đăng ký học phần | Xem học phần, đăng ký học phần và lịch học |
| Đổi mật khẩu | Sinh viên có thể đổi mật khẩu của tài khoản |
| Tra cứu công nợ | Xem công nợ cần phải đóng |
| **Quản trị viên** | Quản lý sinh viên | Cho phép thêm, xóa thông tin của sinh viên, xem danh sách sinh viên |
| Quản lý khoa | Cho phép thêm, xóa, sửa thông tin của khoa, xem danh sách khoa |
| Quản lý chuyên ngành | Cho phép thêm, xóa, sửa thông tin của chuyên ngành, xem danh sách chuyên ngành |
| Quản lý lớp học phần | Cho phép thêm, xóa, sửa thông tin của lớp, xem danh sách lớp phần |
| Quản lý giảng viên | Cho phép thêm, xóa, sửa thông tin của giáo viên, xem danh sách giảng viên |
| Quản lý môn học | Cho phép thêm, xóa, sửa thông tin của môn học, xem danh sách môn học |
| Quản lý học kỳ | Cho phép thêm, xóa, sửa thông tin của học kỳ, xem danh sách học kỳ |
| Quản lý học phần | Cho phép thêm, xóa thông tin của học phần, xem danh sách học phần |

Trong mô hình use-case này có hai actor chính: sinh viên và quản trị viên. Use-case tổng quát và phức tạp về nghiệp vụ, yêu cầu nhiều xử lý cho hệ thống bao gồm: Đăng Nhập, Đăng Ký Học Phần,

### Usecase tổng quát



*Hình 3.1.1 Usecase tổng quát*

### Danh sách các tình huống hoạt động (Use cases)

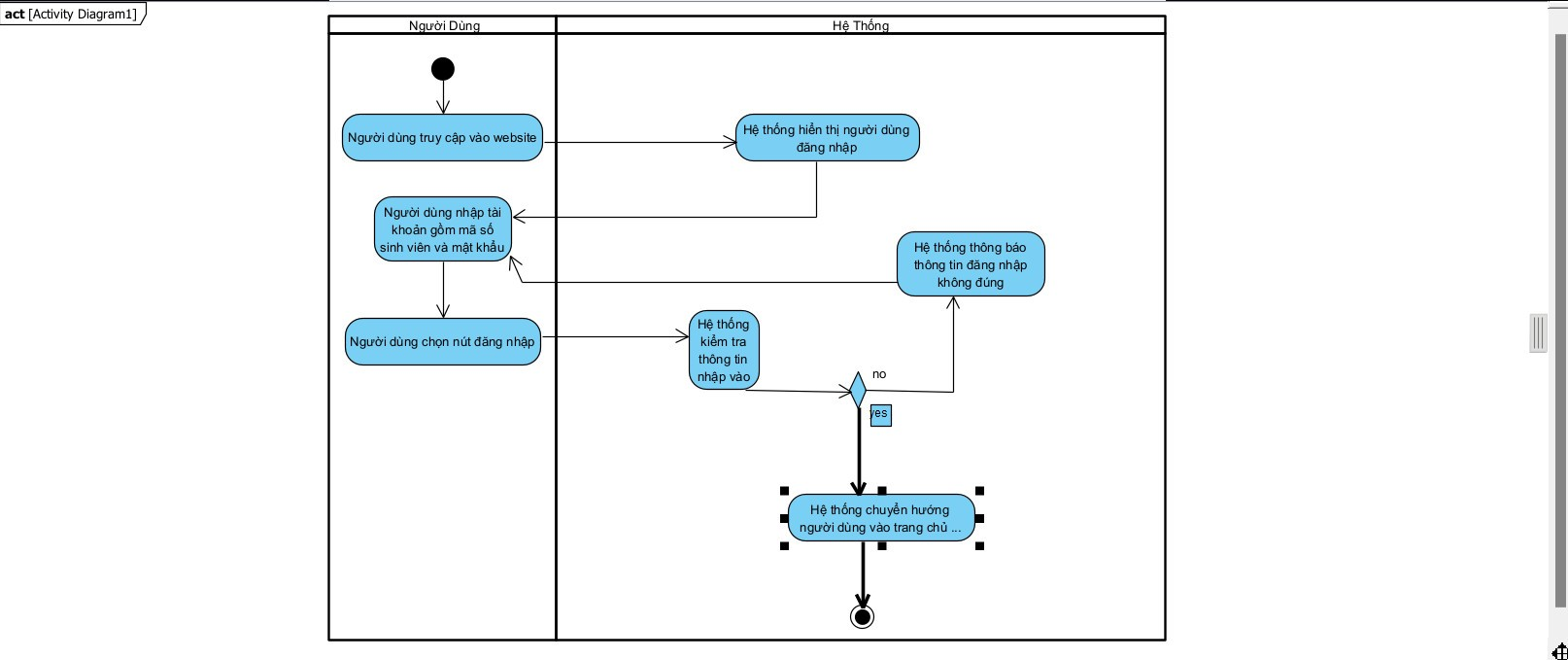
|  |  |
| --- | --- |
| **ID use case** | **Tên Use case** |
| **UC01** | UC01\_Đăng nhập |
| **UC02** | UC02\_Đăng ký học phần |

### Tình huống hoạt động

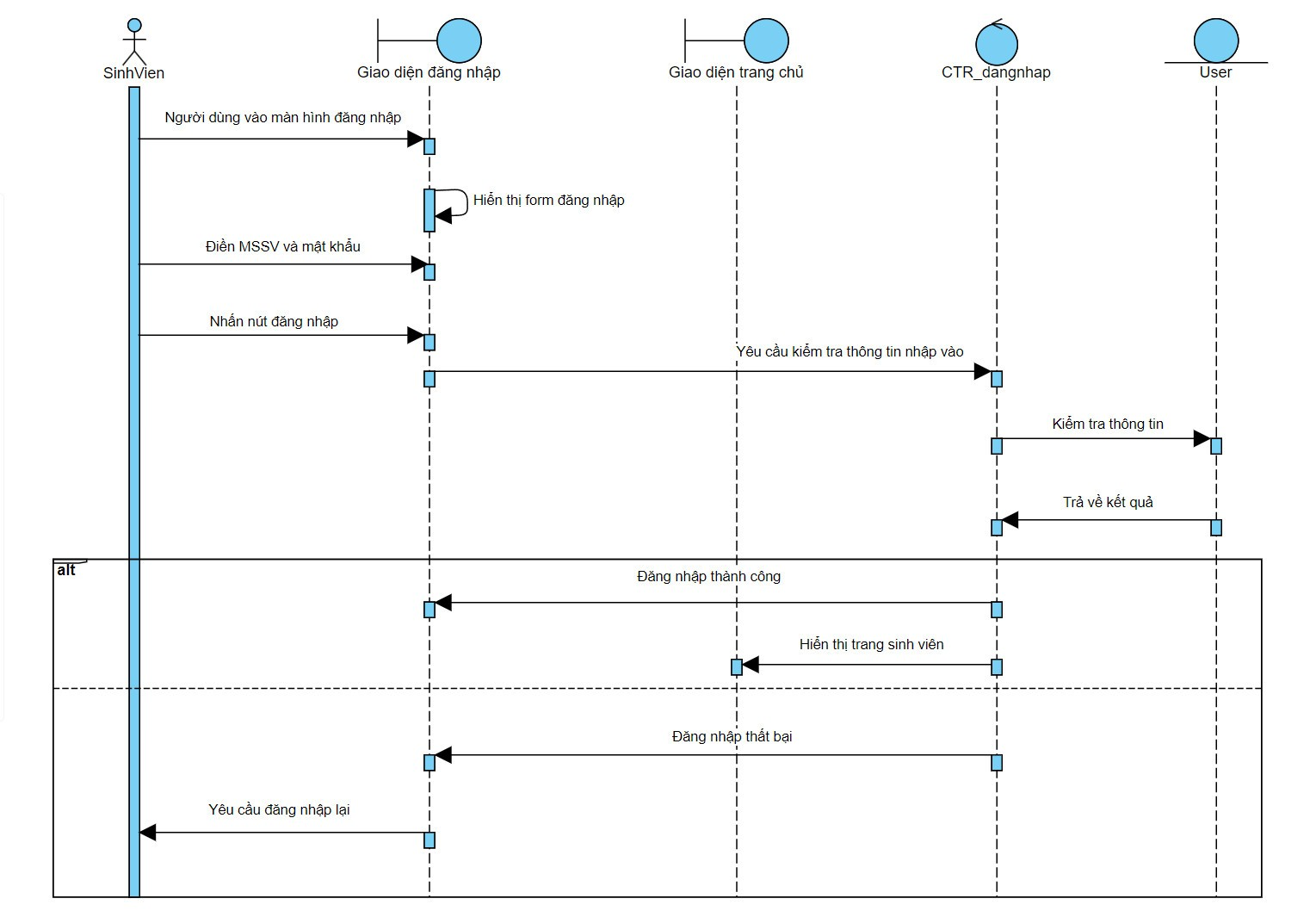
#### UC01\_Đăng nhập

|  |  |
| --- | --- |
| * **Tên use case**: UC01\_Đăng nhập | |
| * **Mô tả sơ lược**: Người dùng đăng nhập vào hệ thống | |
| * **Actor chính**: User | |
| * **Actor phụ**: không có | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng vào trang đăng nhập | 1. Hệ thống hiển thị giao diện nhập thông tin tài khoản |
| 1. Người dùng nhập tài khoản gồm mã số sinh viên và mật khẩu |  |
| 1. Chọn nút đăng nhập | 1. Hệ thống kiểm tra thông tin nhập vào |
|  | 1. Hệ thống chuyển hướng người dùng vào giao diện trang chủ. |
| * **Luồng sự kiện thay thế (alternative flow):** | |
|  | 5.1. Hệ thống thông báo thông tin tài khoản đăng nhập không đúng |
|  | 5.2. Hệ thống quay lại bước 3 |

Activity diagram:



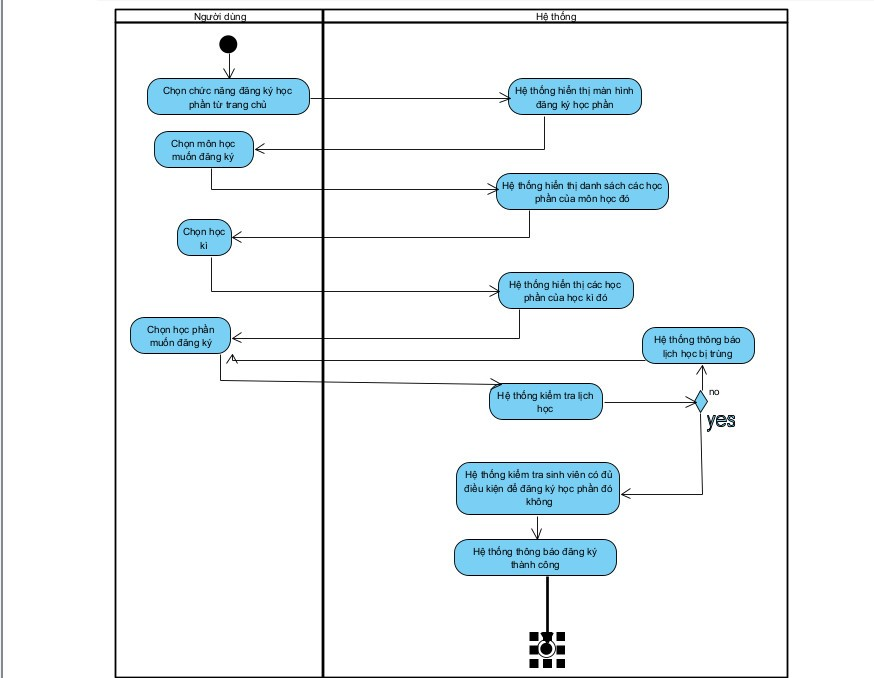
Sequence diagram:



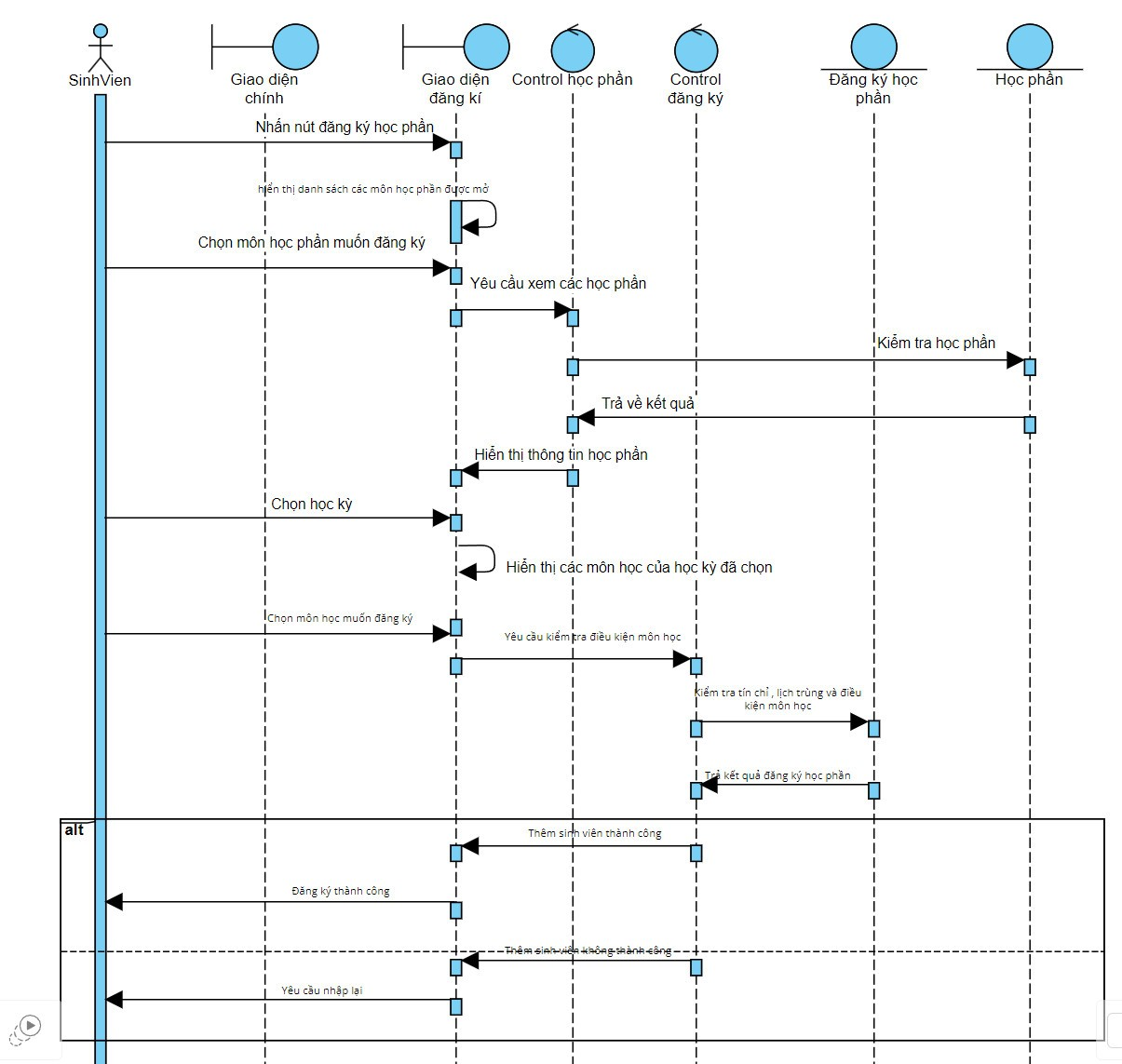
#### UC02\_Đăng ký học phần

|  |  |
| --- | --- |
| * **Tên use case**: UC02\_Đăng ký học phần | |
| * **Mô tả sơ lược**: Sinh viên đăng ký học phần mà mình chọn | |
| * **Actor chính**: Sinh viên | |
| * **Actor phụ**: không có | |
| * **Tiền điều kiện (Pre-condition):** Sinh viên phải đăng nhập vào hệ thống | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Sinh viên chọn môn học muốn đăng ký | 1. Hệ thống hiển thị danh sách các học phần của môn học đó |
| 1. Sinh viên chọn học kì | 1. Hệ thống hiển thị các học phần của học kì đó |
| 1. Sinh viên chọn học phần muốn đăng ký | 1. Hệ thống kiểm tra lịch học |
|  | 1. Hệ thống kiểm tra sinh viên có đủ điều kiện để đăng ký học phần đó không |
|  | 1. Hệ thống thông báo đăng ký thành công |
| * **Luồng sự kiện thay thế (alternative flow):** | |
|  | 6.1 Hệ thống thông báo lịch học bị trùng |
|  | 6.2 Hệ thống quay lại bước 5 |

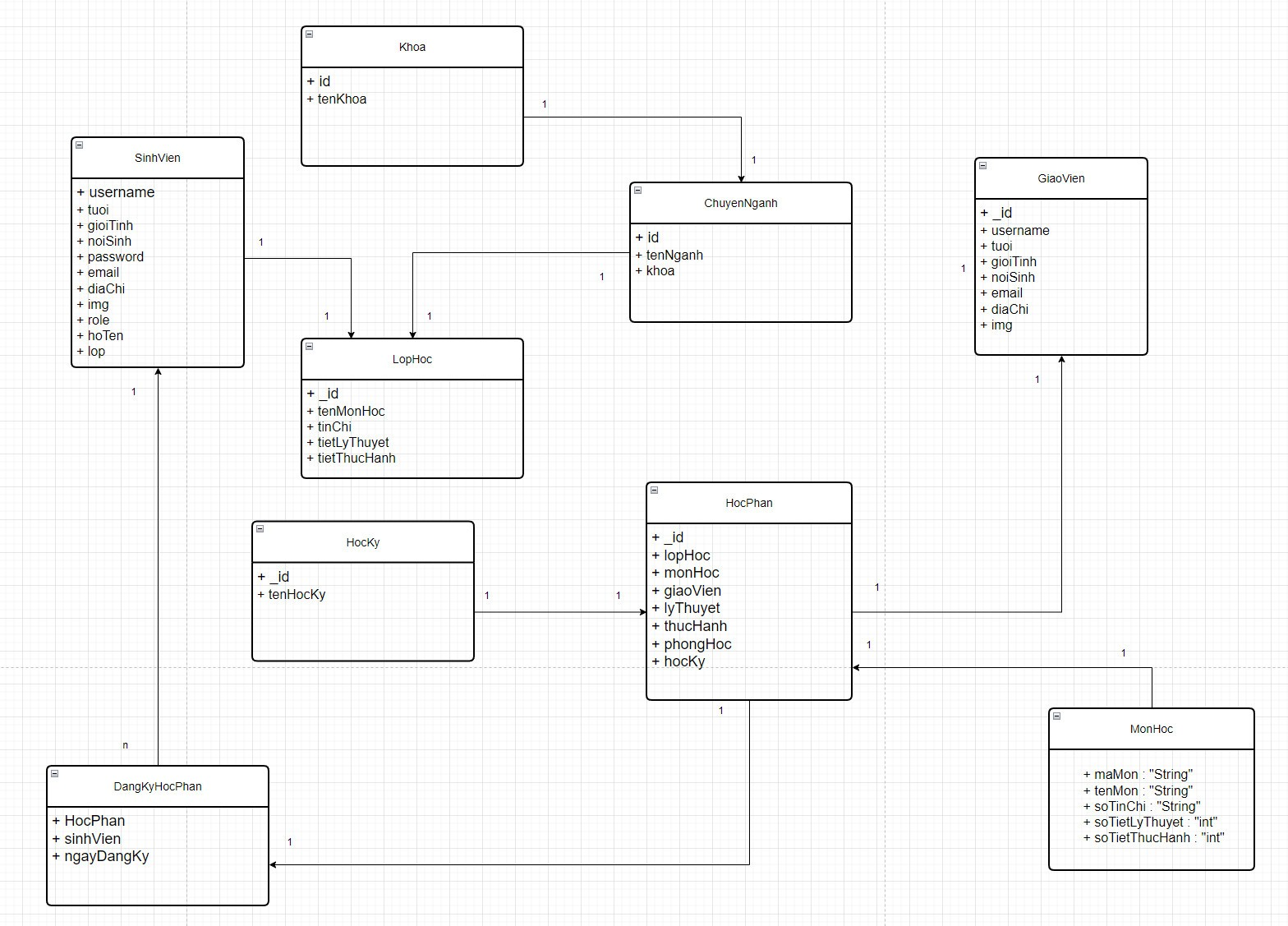
Activity diagram:



Sequence diagram:



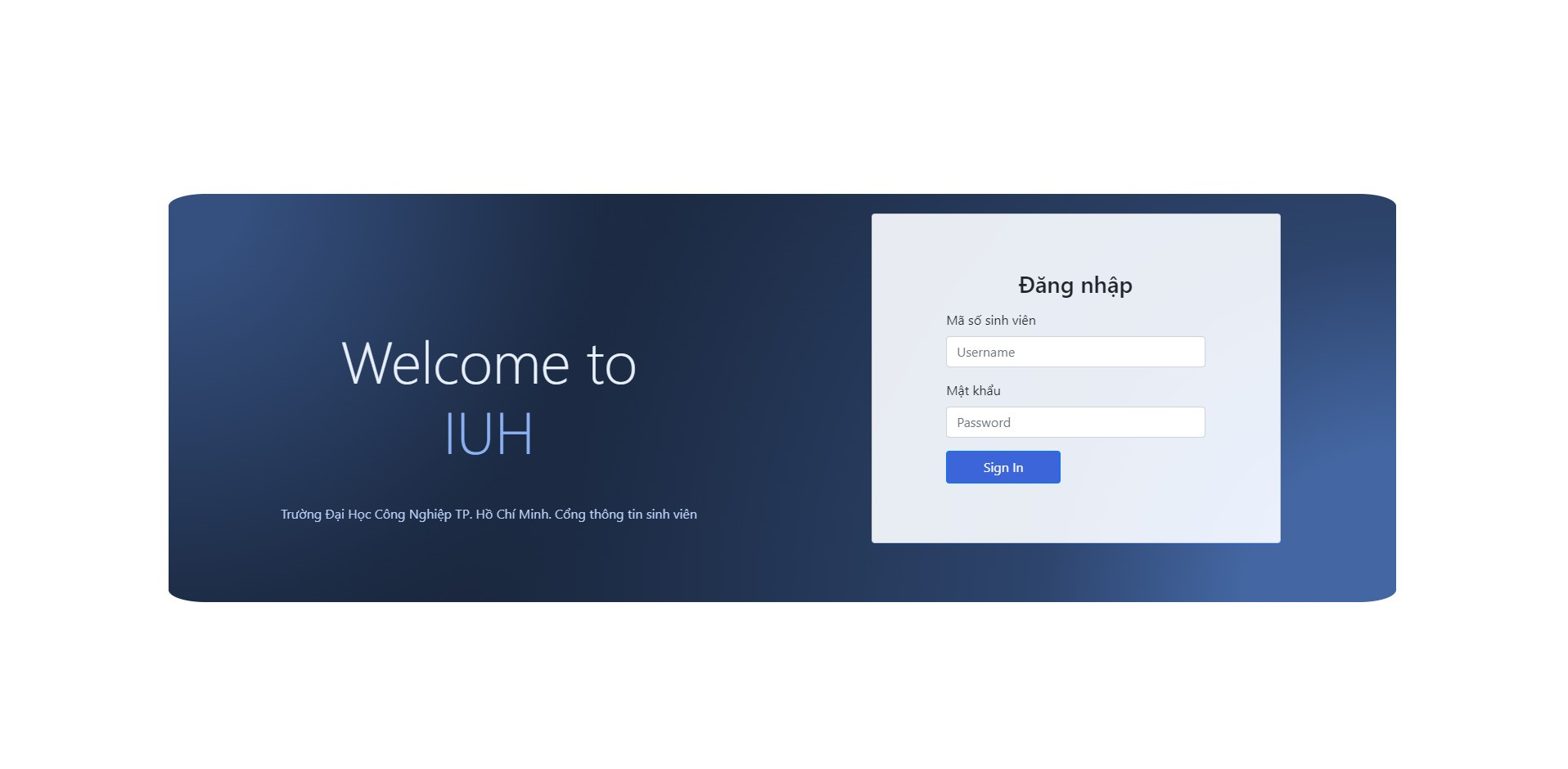
## Class diagram



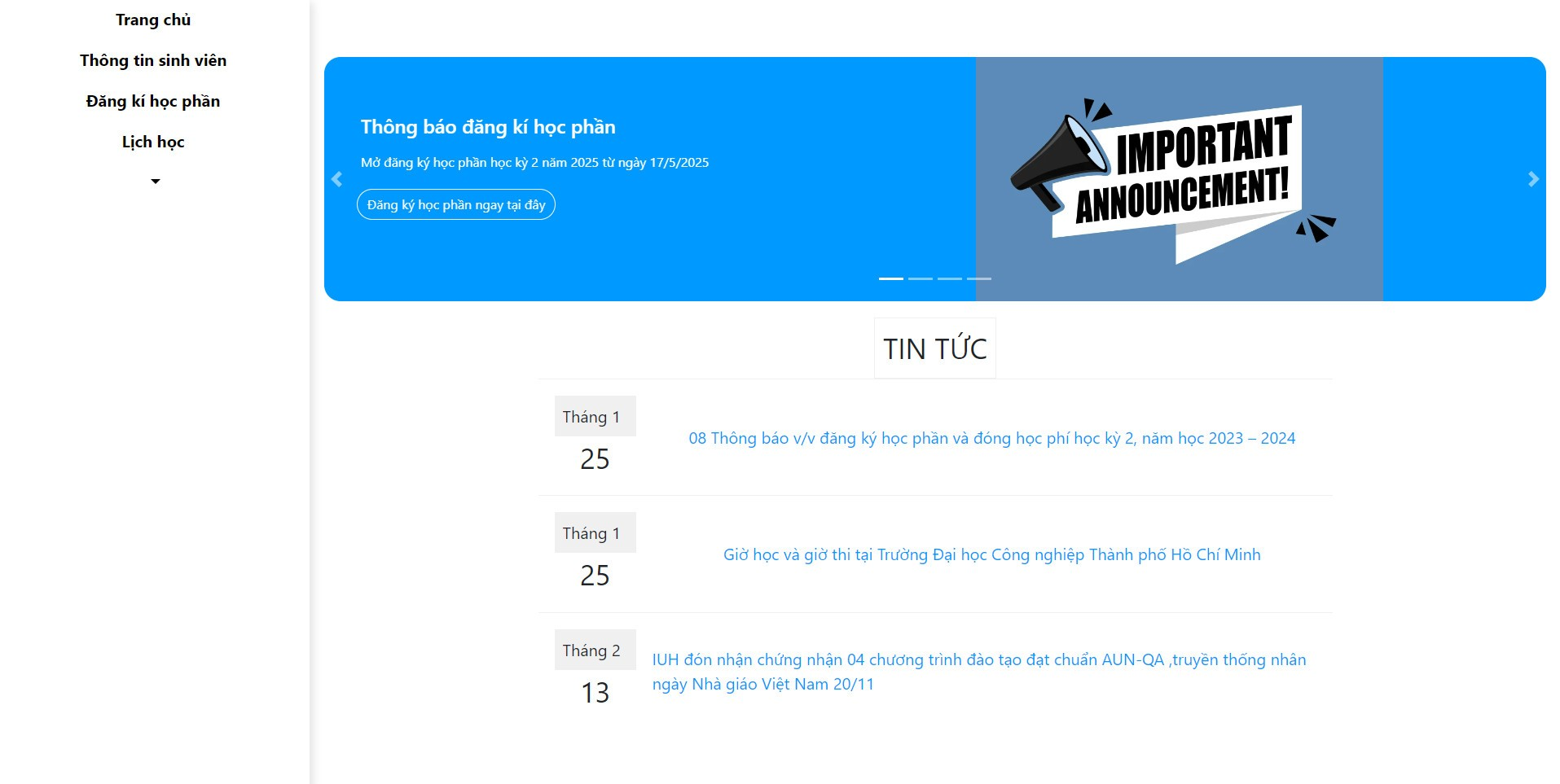
# : HIỆN THỰC

## Giao diện của hệ thống

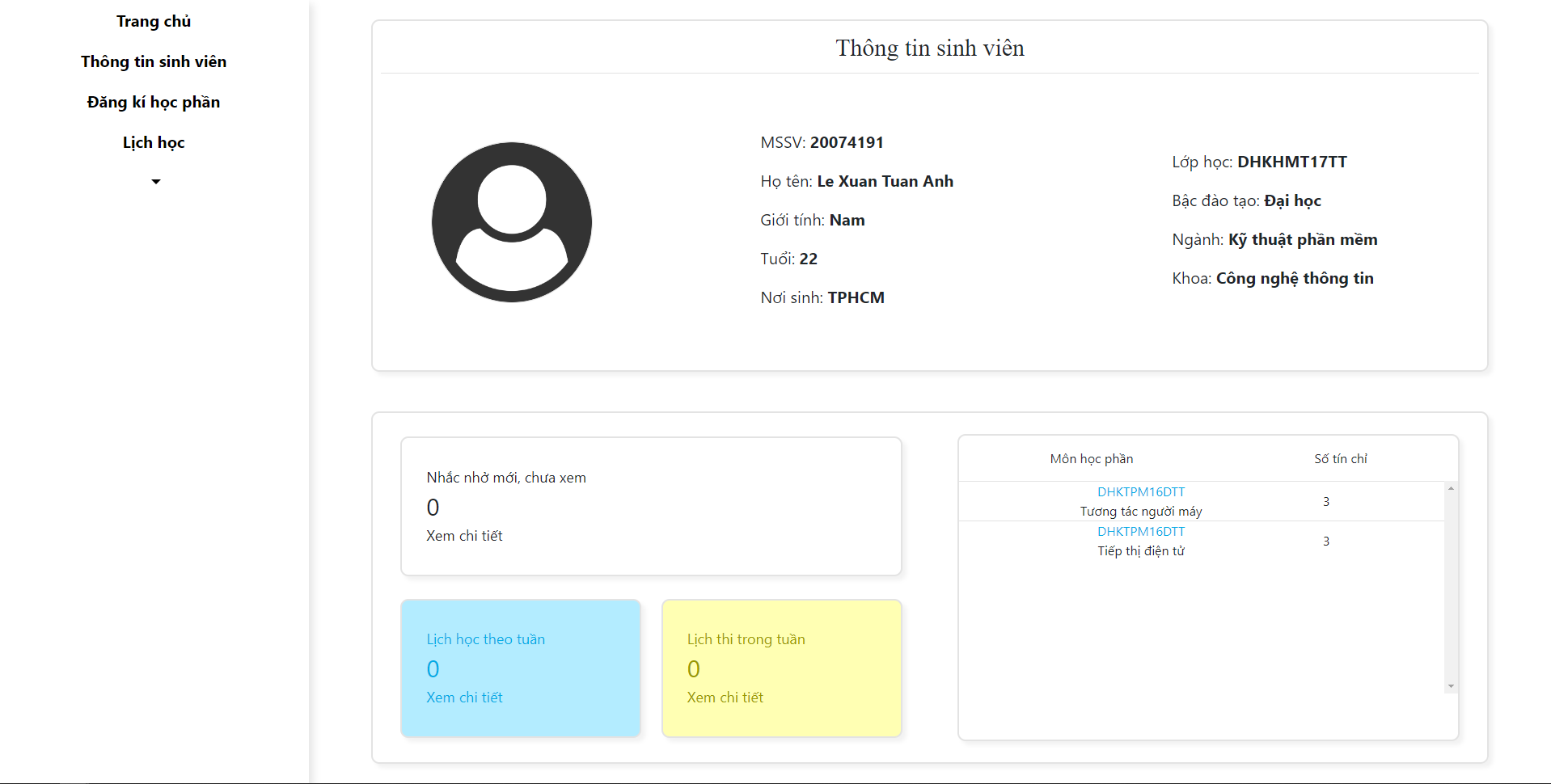
### Giao diện web



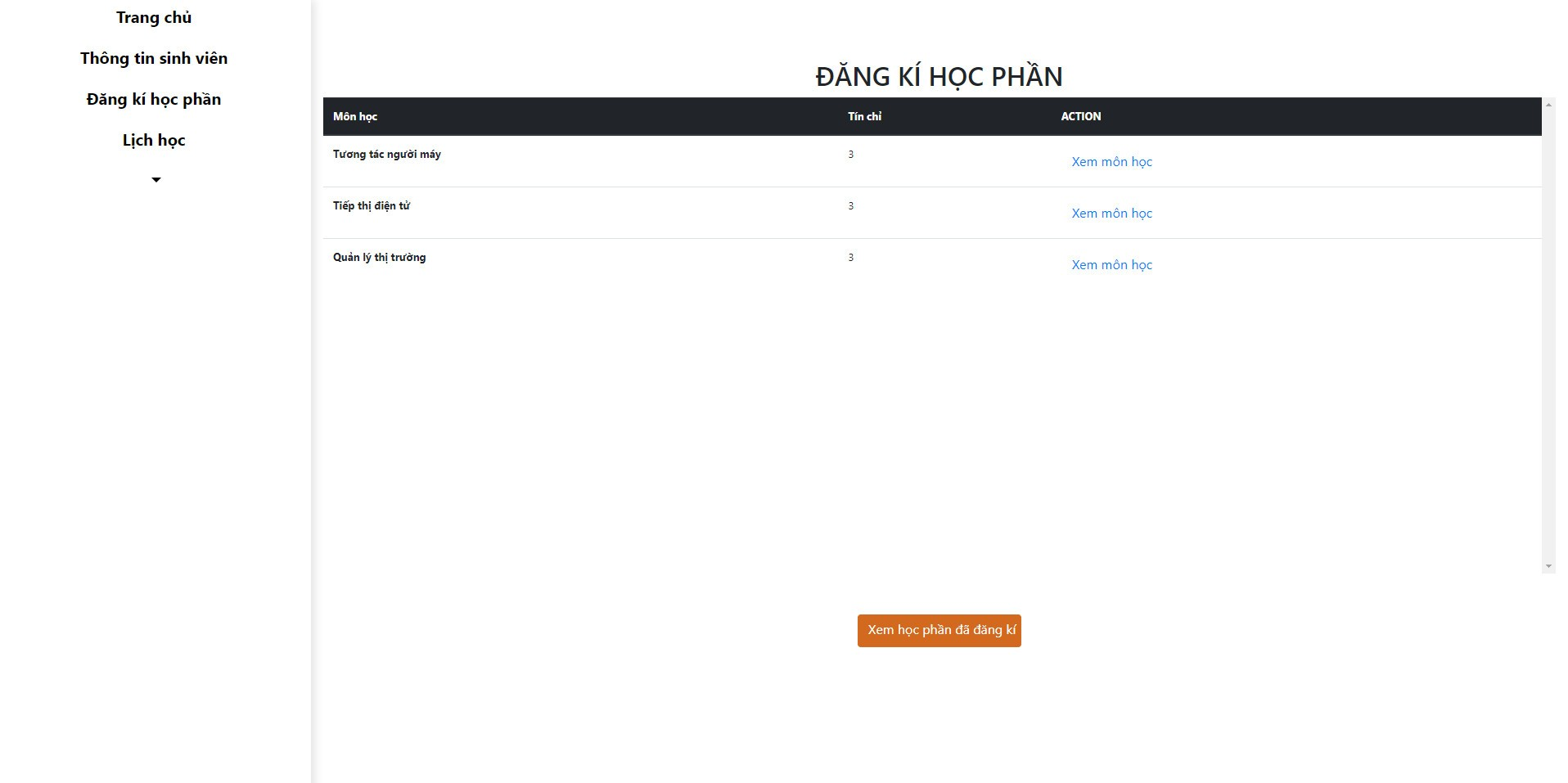
*Hình 4.1.1.1 Màn hình giao diện đăng ký*

**

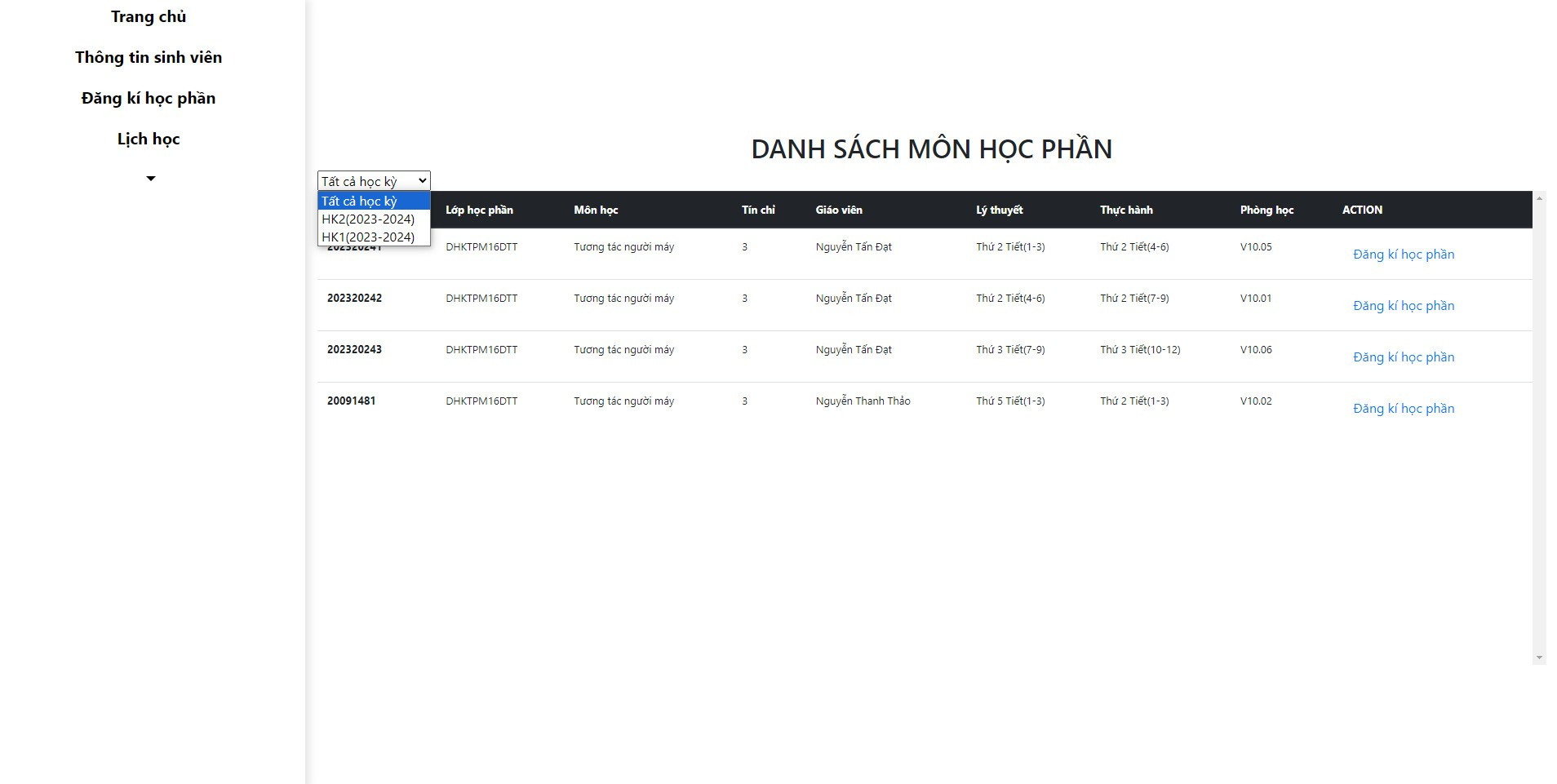
*Hình 4.1.1.2 Màn hình giao diện trang chủ của sinh viên*

**

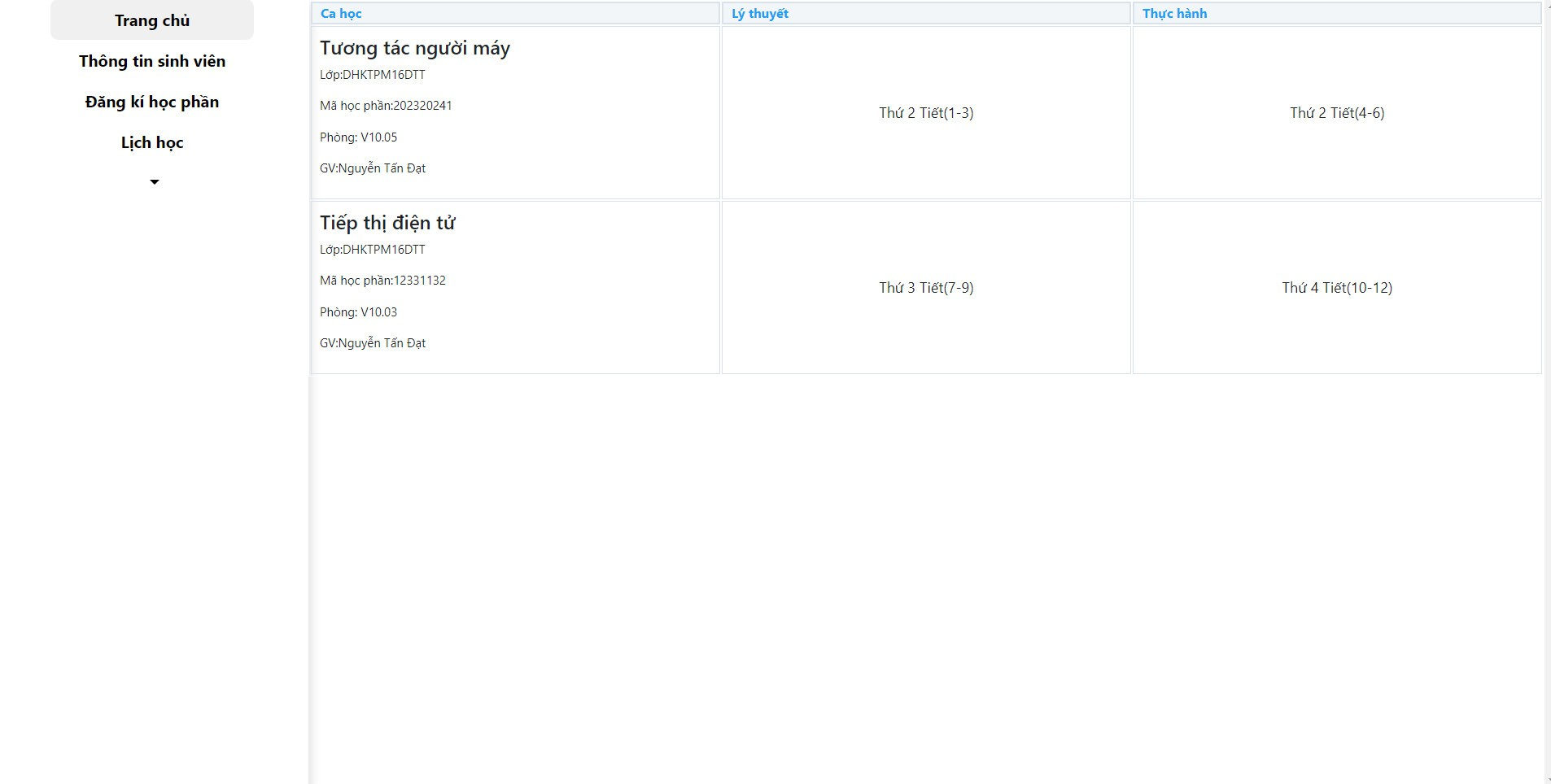
*Hình 4.1.1.3 Màn hình giao diện thông tin sinh viên*

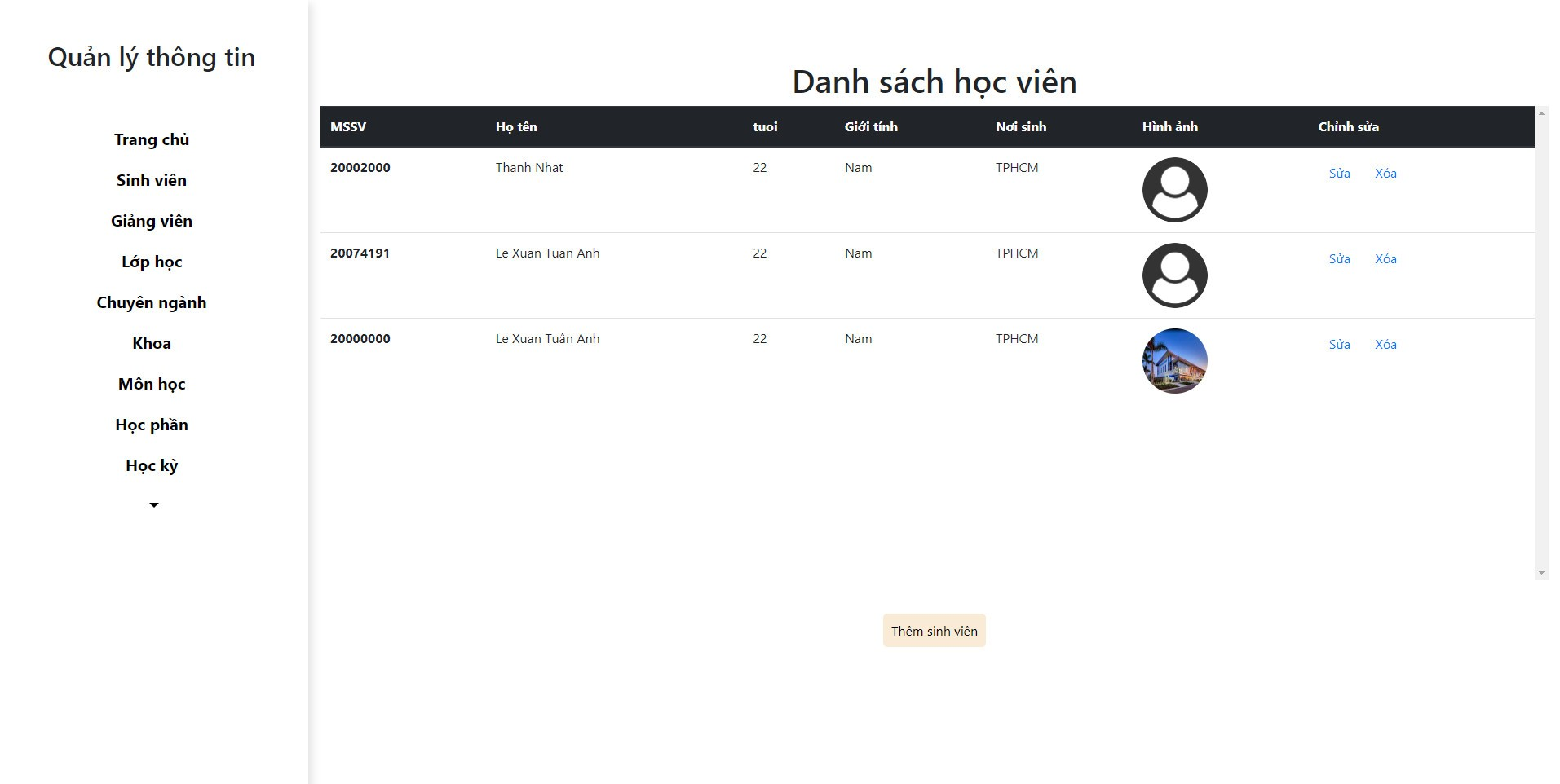
**

*Hình 4.1.1.4 Màn hình giao diện đăng ký học phần*

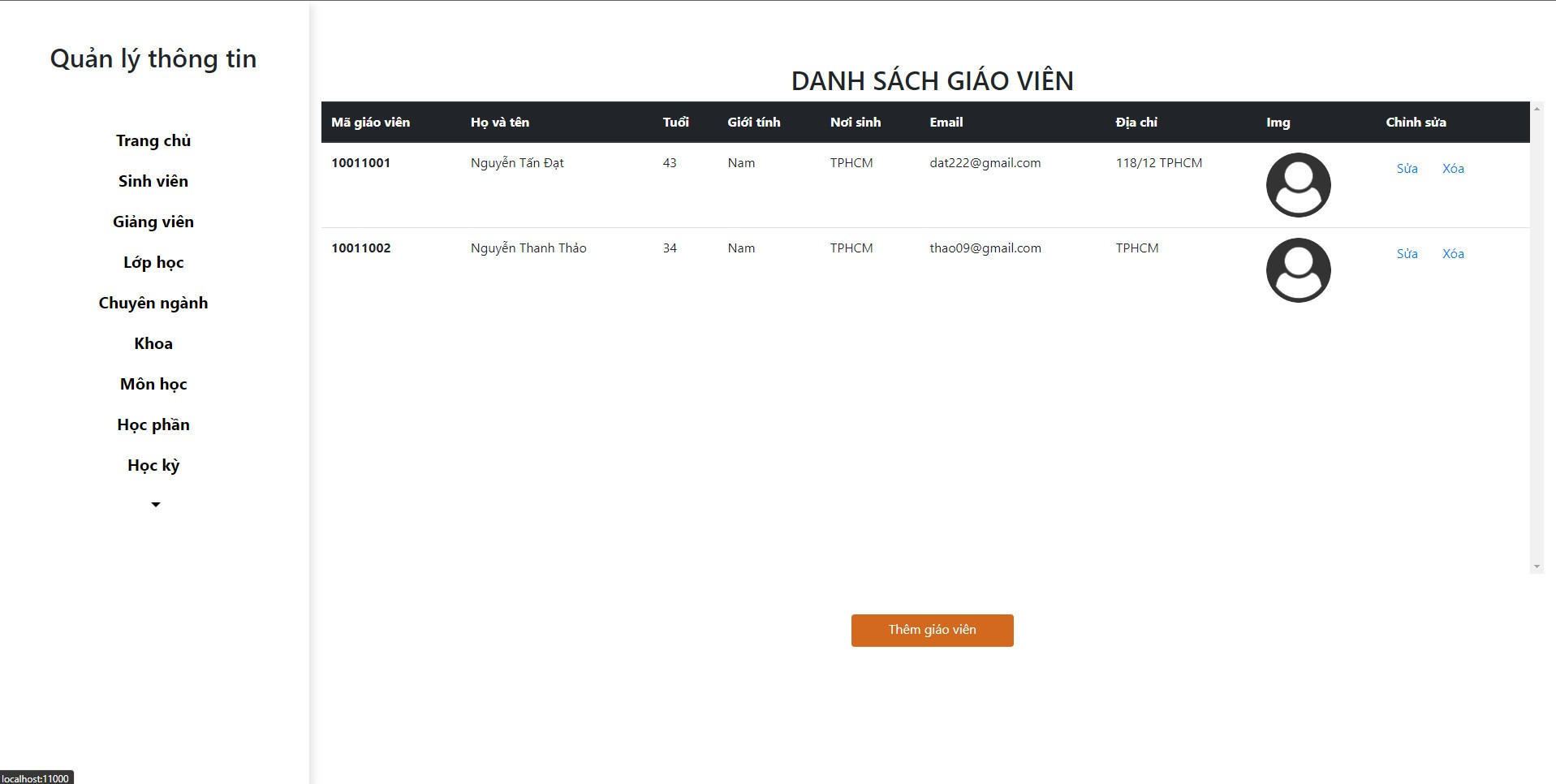
**

*Hình 4.1.1.5 Màn hình danh sách học phần*

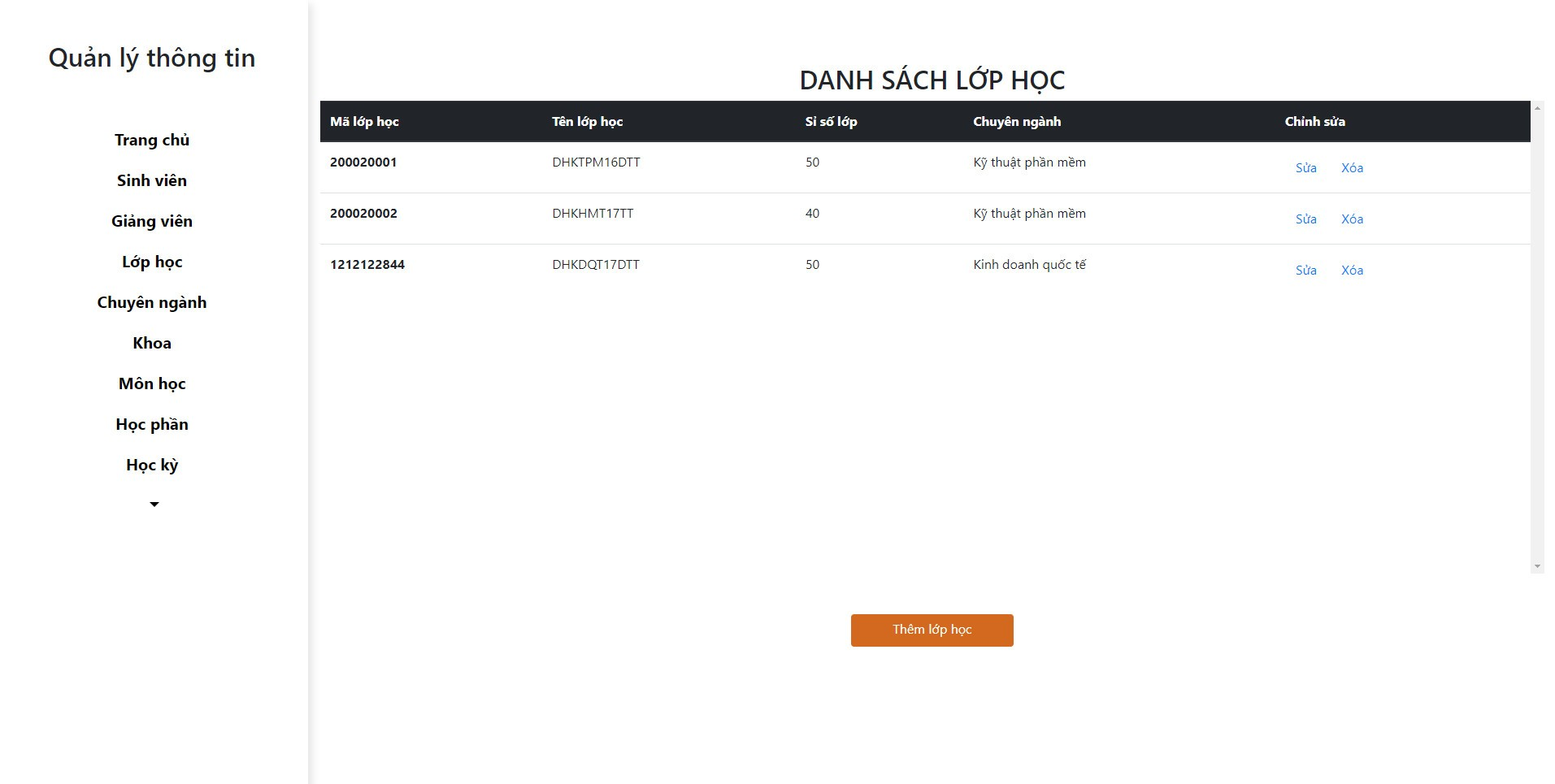
*  
Hình 4.1.1.6 Màn hình lịch học*

**

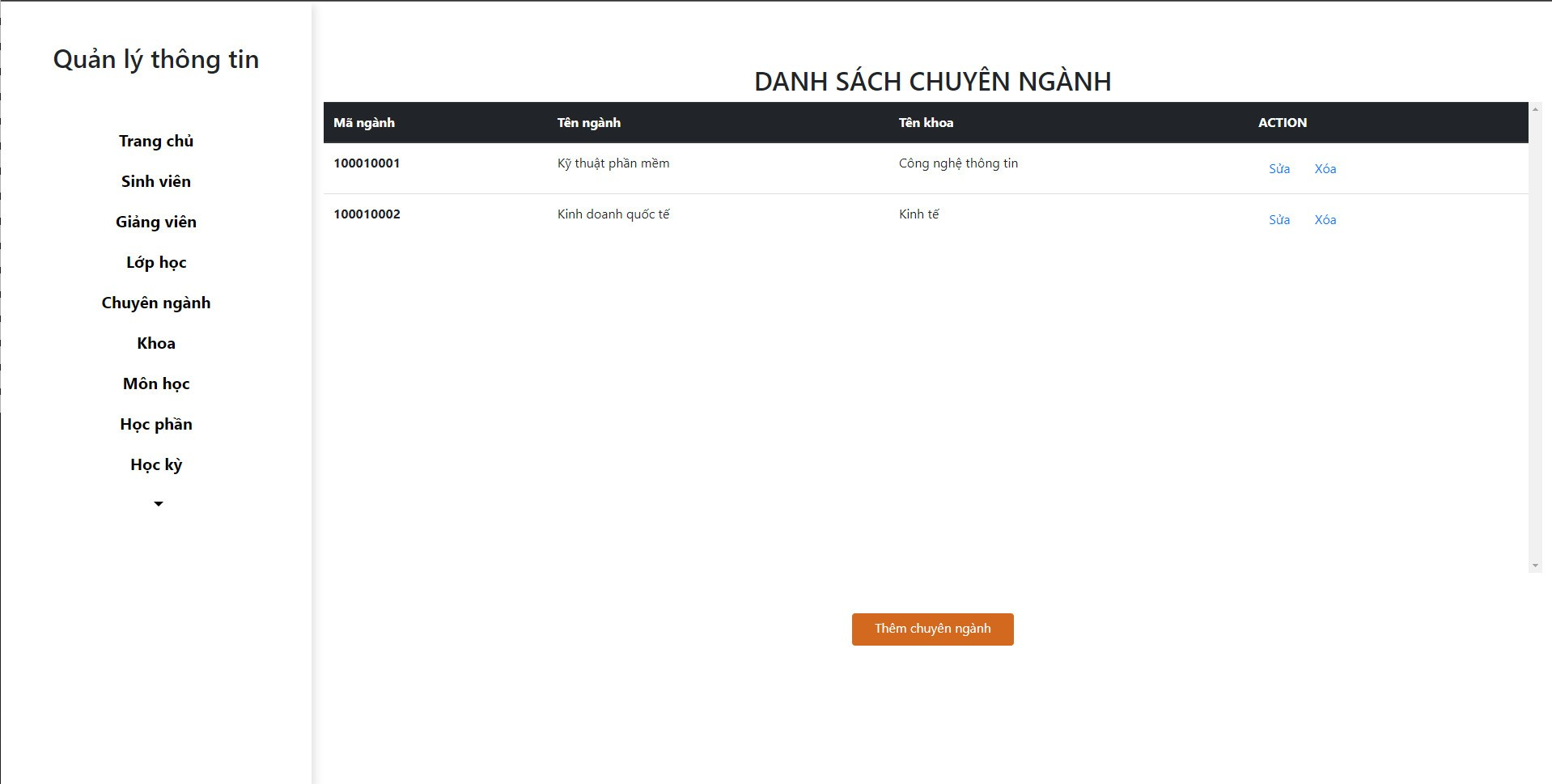
*Hình 4.1.1.7 Màn hình danh sách sinh viên*

**

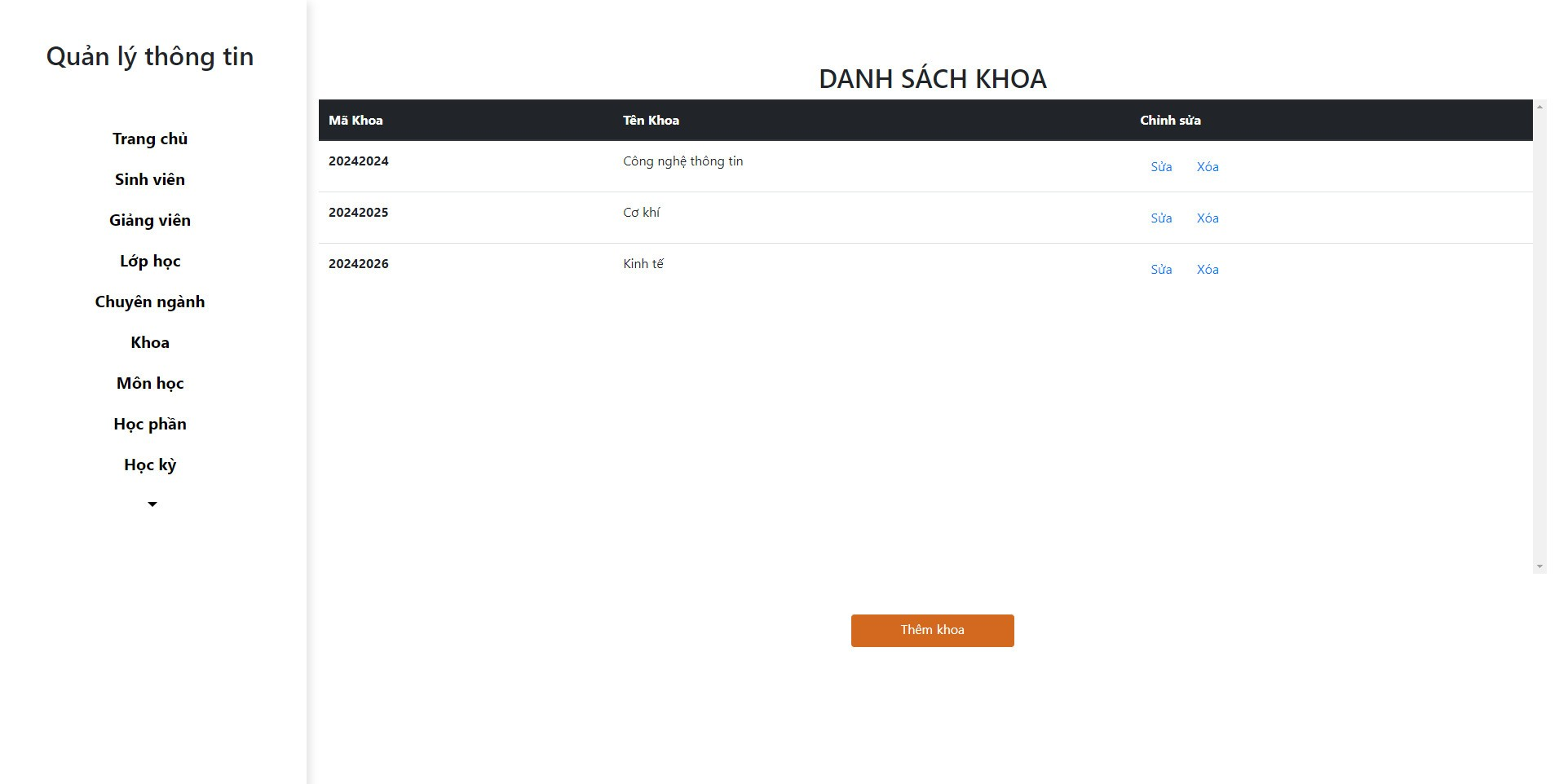
*Hình 4.1.1.8 Màn hình danh sách giảng viên*

**

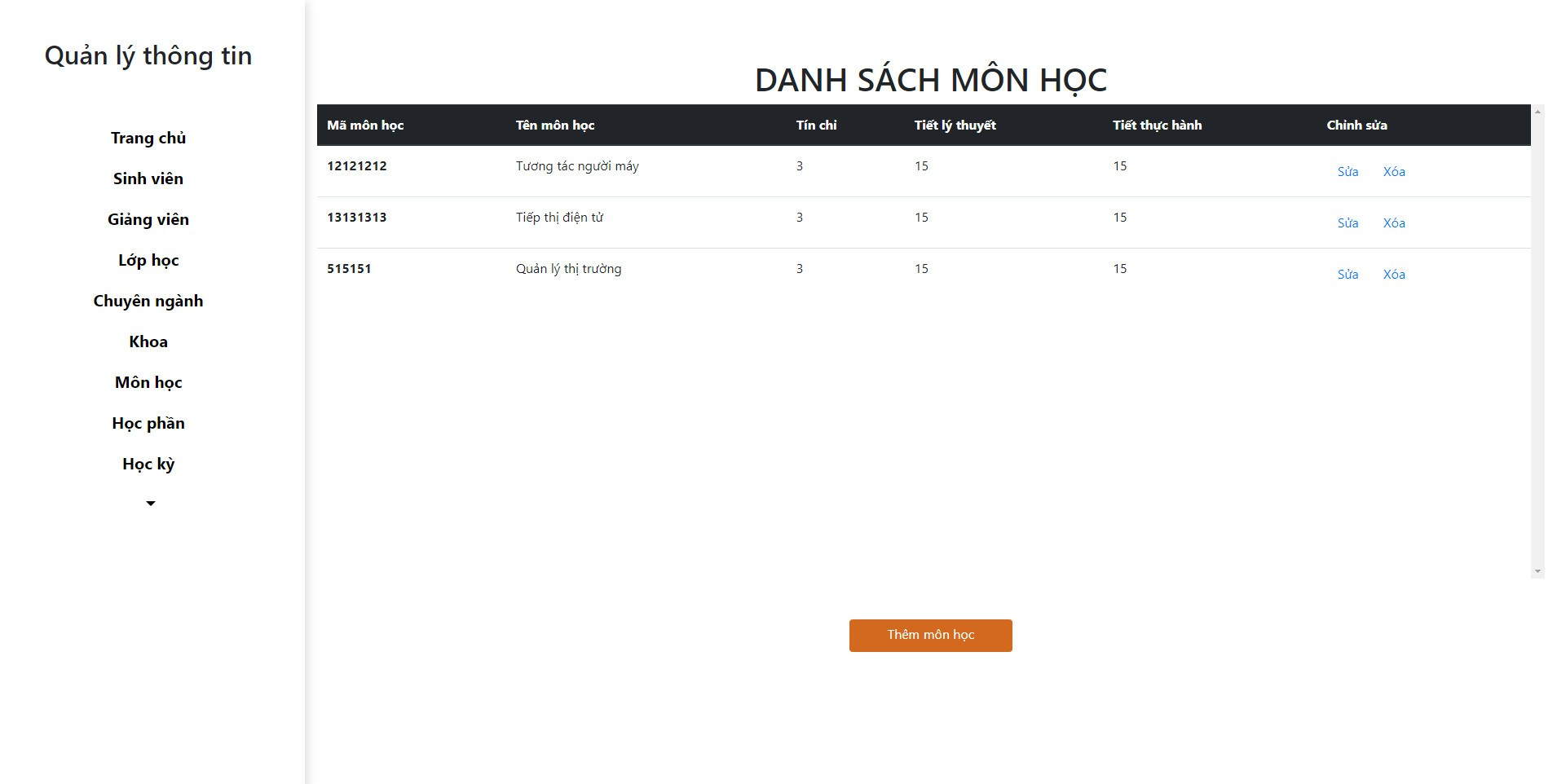
*Hình 4.1.1.9 Màn hình danh sách lớp học*

**

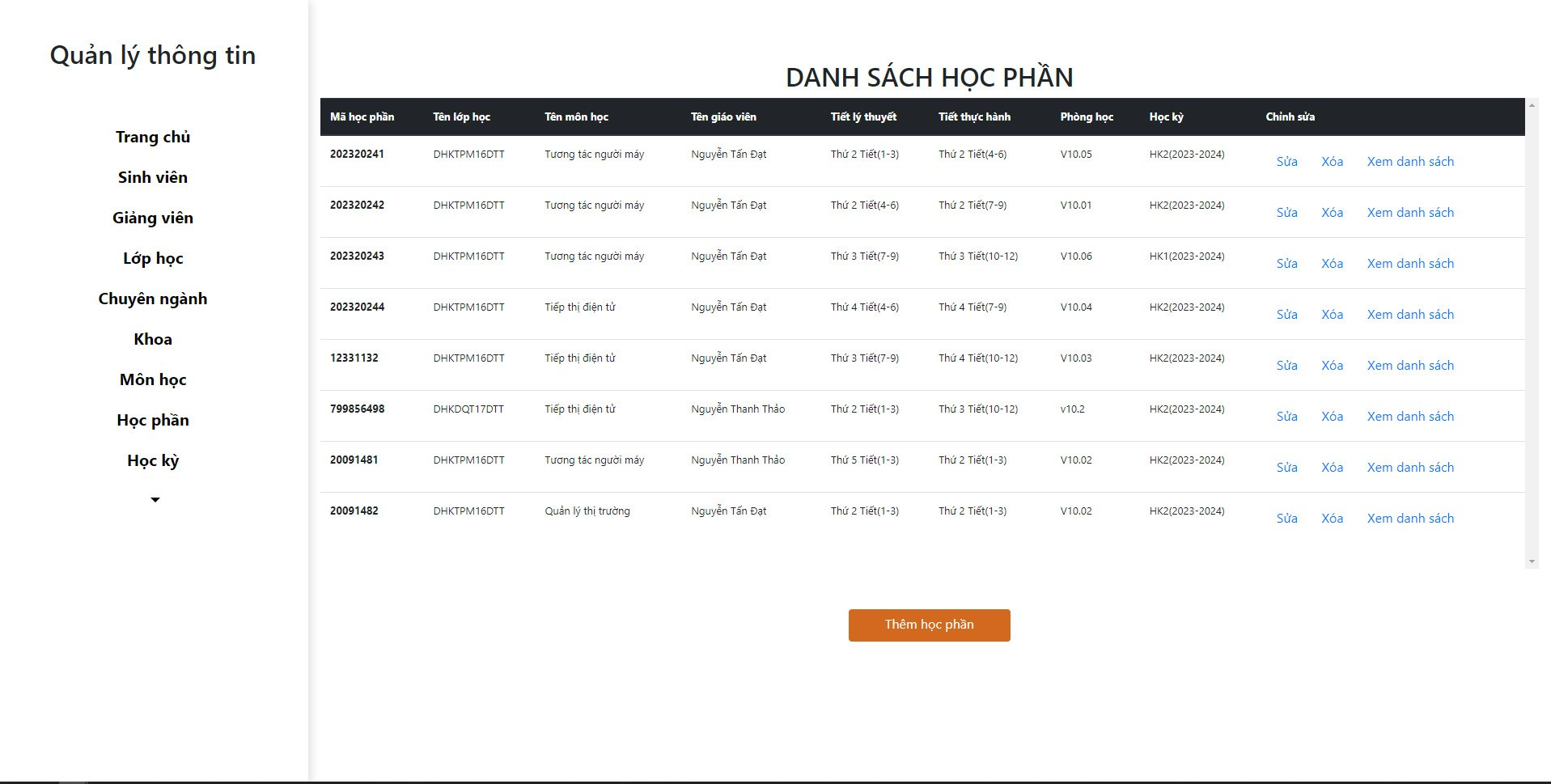
*Hình 4.1.1.10 Màn hình danh sách chuyên ngành*

**

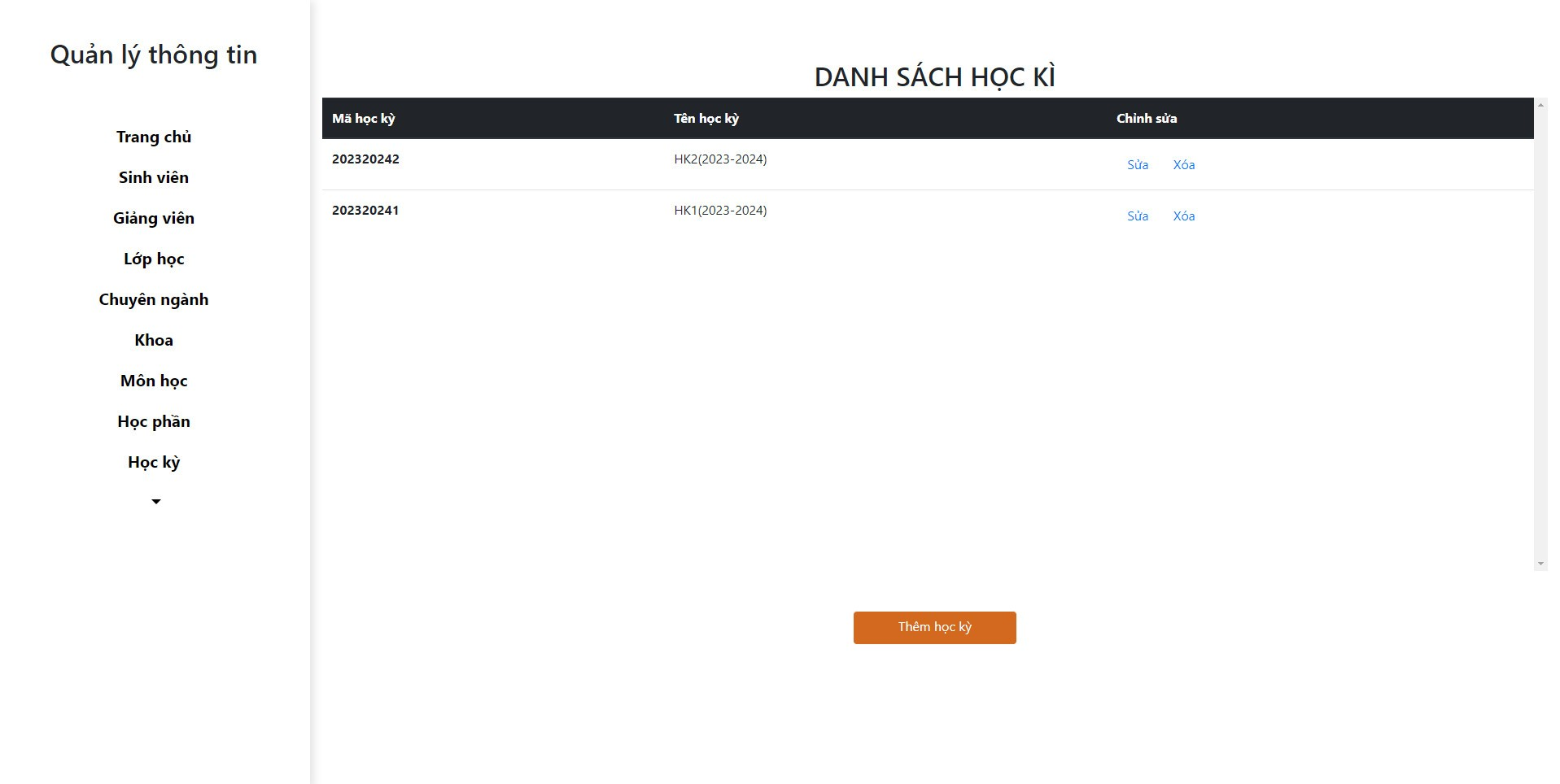
*Hình 4.1.1.10 Màn hình danh sách khoa*

**

*Hình 4.1.1.10 Màn hình danh sách môn học*

**

*Hình 4.1.1.10 Màn hình danh sách học phần*

**

*Hình 4.1.1.10 Màn hình danh sách học kì*

# : KẾT LUẬN

## Kết quả đạt được

Dự án đã có thể đưa vào hoạt động với một vài chức năng cơ bản

* Xem thông tin cá nhân
* Quản lý đăng ký học phần
* Tra cứu công nợ

Đối với phần của quản trị viên thì đã tích hợp được nhiều chức năng như là

* Quản lý khoa, chuyên ngành, sinh viên, môn học, học phần, lớp học phần, giảng viên, học kì

## Hạn chế của đồ án

* Vẫn còn hạn chế một vài chức năng như xem tiến trình học tập, chương trình khung, đăng ký xét tốt nghiệp, lịch theo tiến độ.
* Phần giao diện cần phải cải thiện nhiều hơn để nhìn tối ưu hơn.
* Thiếu kinh phí để tiếp tục phát triển dự án

## Hướng phát triển

* Hoàn thiện những chức năng còn thiếu, tiếp tục phát triển phần giao diện.
* Tích hợp chức năng học tập trực tuyến

# TÀI LIỆU THAM KHẢO

Các tài liệu Tiếng Anh

1. Kevin Faaborg & Sandro Pasquali - Mastering Node.js Second Edition – December 2017
2. David Herron, Diogo Resende & Valentin Bojinov - Node.js Complete Reference Guide – December 2018

Các tài liệu từ Internet

1. <https://nodejs.org/en/docs/>
2. <https://expressjs.com/en/guide/routing.html>
3. <https://www.mongodb.com/docs/manual/reference/operator/aggregation/graphLookup/>
4. <https://aws.amazon.com/vi/what-is/javascript/>
5. <https://viblo.asia/p/ap-dung-kien-truc-3-layer-architecture-vao-project-nodejs-RQqKLqEmZ7z>