Trường Đại học Bách Khoa Hà Nội 

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Hệ thống Quản lý sinh viên

ứng dụng Microsoft .NET WEB API

Tạ Tô Minh Chí

Hà Nội, 06/2018

Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Hệ thống Quản lý sinh viên

ứng dụng Microsoft .NET WEB API

|  |  |
| --- | --- |
| Sinh viên thực hiện | Tạ Tô Minh Chí |
| Người hướng dẫn | TS. Bùi Thị Mai Anh |

Hà Nội, 06/2018

# Lời cam kết

Họ và tên sinh viên: Tạ Tô Minh Chí

Điện thoại liên lạc: 01696510249 Email: taminhchi96@gmail.com

Lớp: CN-CNTT 02 K59 Hệ đào tạo: Chính quy

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

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 28 tháng 5 năm 2018*  Tác giả ĐATN  *Tạ Tô Minh Chí* Nhận xét của giáo viên hướng dẫn ……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………… |

# Lời cảm ơn

Trong thời gian làm đồ án tốt nghiệp, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè.  Với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến tất cả các cá nhân và cơ quan đã tạo điều kiện giúp đỡ trong quá trình học tập và hoàn thành đồ án.

Em xin cảm ơn TS. Bùi Thị Mai Anh – GV Bộ môn Công nghệ phần mềm đã tận tình hướng dẫn em trong suốt quá trình làm Đồ án Tốt Nghiệp này.

Em xin cảm ơn TS. Nguyễn Thị Thanh Tú – GV Bộ môn Công nghệ phần mềm đã hỗ trợ, giúp đỡ em trong suốt quá trình làm đồ án

Em xin cảm ơn anh Nguyễn Văn Mạnh – chuyên viên Ban Nghiên cứu và Phát triển – Trung tâm Phát triển phần mềm - Cty Cổ phần MISA đã hướng dẫn em về công nghệ trong suốt quá trình làm đồ án

Xin cảm ơn Ban Nghiên cứu và Phát triển – Trung tâm Phát triển phần mềm - Cty Cổ phần MISA đã tạo điều kiện cho em được thực tập và học hỏi về các công nghệ trong suốt quá trình làm đồ án.

Xin cảm ơn gia đình, bạn bè đã động viên, quan tâm và là nguồn động lực lớn lao giúp em trong quá trình làm đồ án.

# Tóm tắt

DATN của em có nội dung về công nghệ Microsoft .NET API và ứng dụng nó trong hệ thống quản lý sinh viên. Thực trạng hiện tại, hệ thống quản lý sinh viên của nhiều trường Đại học còn sử dụng các công nghệ cũ như MVC, khiến cho việc phục vụ các bạn sinh viên truy cập không đạt hiệu quả cao, tình trạng tắc nghẽn, sập server diễn ra thường xuyên. Công nghệ .NET API là một công nghệ mới của Microsoft, giải quyết tốt vấn đề cân tải ở hệ thống cũng như hệ thống đa nền tảng. Trên thực tế nó đã được áp dụng trong hệ thống quản lý trường học của Cty MISA và đạt hiệu quả cao. Với mong muốn giải quyết thực trạng của các hệ thống quản lý sinh viên hiện nay, em quyết định sử dụng công nghệ này trong đồ án của mình nhằm xây dựng một hệ thống quản lý sinh viên tốt hơn cũng như mở rộng ra để giải quyết các bài toán khác như tra cứu điểm thi đại học, thương mại điện tử, các ứng dụng đa nền tảng. Giúp cho công việc quản lý của các các bộ và thầy cô được thuận lợi và đơn giản hơn, giúp các bạn sinh viên có một hệ thống phục vụ nhanh hơn, thuận tiện hơn để đạt hiệu quả cao trong học tập. Cùng với đó là ứng dụng rộng rãi trong các lĩnh vực khác trong đời sống, như là hệ thống tra cứu điểm thi đại học quá tải mỗi mùa tuyển sinh, các ứng dụng đa nền tảng chạy được cả trên web, desktop và mobile, …

# Abstract

The main content of my thesis is about Microsoft .NET API technology and its application in the student management system. at the moment, student management system of many universities still using old technology such as MVC, which makes the connection of the student to the system not so stability, congestion, and the server crashed usually occurs when there is too many student access at the same time. Therefore, the needed of the new technology for the management system is the important thing we have to consider .NET API is a new technology from Microsoft that solves both two problem is system load balancing but the cross-platform issues. In fact, it has been applied in the student management of MISA JSC and has achieved high efficiency. With a goal is to solve the issues in all student management system, I decided to bring this technology to my thesis to build a better management system as well as expand it to solve the other problems like college entrance exam scores, e-commerce, cross-platform applications. This will help the management of the ministries and teachers become easier and simple, and also help students have a faster service system, more convenient to achieve high academic performance.

# Mục lục

1. **Contents**

[**Lời cam kết iii**](#_gjdgxs)

[**Nhận xét của giáo viên hướng dẫn iv**](#_30j0zll)

[**Lời cảm ơn v**](#_1fob9te)

[**Tóm tắt vi**](#_3znysh7)

[**Abstract vii**](#_2et92p0)

[**Mục lục viii**](#_tyjcwt)

[**Danh mục hình vẽ ix**](#_3dy6vkm)

[**Danh mục bảng x**](#_1t3h5sf)

[**Danh mục các từ viết tắt xii**](#_4d34og8)

[**Danh mục thuật ngữ xiii**](#_2s8eyo1)

[**Chương 1 Giới thiệu đề tài 1**](#_3rdcrjn)

[**1.1 Đặt vấn đề 1**](#_26in1rg)

[**1.2 Mục tiêu và phạm vi đề tài 2**](#_lnxbz9)

[**1.3 Định hướng giải pháp 3**](#_35nkun2)

[**1.4 Bố cục đồ án 4**](#_1ksv4uv)

[**Chương 2 Khảo sát và phân tích yêu cầu 6**](#_44sinio)

[**2.1 Khảo sát hiện trạng 6**](#_2jxsxqh)

[**2.2 Tổng quan chức năng 8**](#_4i7ojhp)

[2.2.1 Biểu đồ use case tổng quan 8](#_2xcytpi)

[2.2.2 Biểu đồ use case phân rã 10](#_3as4poj)

[2.2.3 Quy trình nghiệp vụ 13](#_147n2zr)

[**2.3 Đặc tả chức năng 16**](#_32hioqz)

[2.3.1 Đặc tả use case Thêm sinh viên 16](#_1hmsyys)

[2.3.2 Đặc tả use case Tìm kiếm sinh viên 19](#_3fwokq0)

[2.3.3 Đặc tả use case Sửa thông tin sinh viên 23](#_19c6y18)

[2.3.4 Đặc tả use case Tìm học phần 26](#_37m2jsg)

[2.3.5 Đặc tả use case đăng ký học phần 30](#_111kx3o)

[2.3.6 Đặc tả use case đăng ký lớp học 33](#_2zbgiuw)

[**2.4 Yêu cầu phi chức năng 37**](#_sqyw64)

[**Chương 3 Công nghệ sử dụng 39**](#_1rvwp1q)

[**3.1 Danh sách các công nghệ sử dụng 39**](#_4bvk7pj)

[**3.2 HTML, CSS, Javascript 40**](#_2r0uhxc)

[**3.3 JQuery và jQueryUI 40**](#_1664s55)

[**3.4 C# và .NET Framework 41**](#_3q5sasy)

[**3.5 RESTful Web Service 42**](#_25b2l0r)

[**3.6 Microsoft .NET Web API 45**](#_43ky6rz)

[**3.7 Repository Pattern 46**](#_xvir7l)

[**3.8 Unit of work 47**](#_1x0gk37)

[**3.9 Unity Dependency Injection 48**](#_1baon6m)

[**Chương 4 Phát triển và triển khai ứng dụng 52**](#_pkwqa1)

[**4.1 Thiết kế kiến trúc 52**](#_39kk8xu)

[4.1.1 Lựa chọn kiến trúc 52](#_1opuj5n)

[4.1.2 Thiết kế tổng quan 54](#_2nusc19)

[4.1.3 Thiết kế chi tiết gói 55](#_3mzq4wv)

[**4.2 Thiết kế chi tiết 57**](#_319y80a)

[4.2.1 Thiết kế giao diện 57](#_1gf8i83)

[4.2.2 Thiết kế lớp 58](#_upglbi)

[4.2.3 Thiết kế cơ sở dữ liệu 61](#_meukdy)

[**4.3 Xây dựng ứng dụng 63**](#_zu0gcz)

[4.3.1 Thư viện và công cụ sử dụng 63](#_3jtnz0s)

[4.3.2 Kết quả đạt được 63](#_4iylrwe)

[4.3.3 Minh hoạ các chức năng chính 64](#_1d96cc0)

[**4.4 Triển khai 66**](#_3x8tuzt)

[**Chương 5 Các giải pháp và đóng góp nổi bật 67**](#_2ce457m)

[**5.1 .NET WEB API được ứng dụng như thế nào trong Project 67**](#_rjefff)

[**5.2 Tại sao phải áp dụng kiến trúc REST? 70**](#_1qoc8b1)

[**5.3 Unit of Work có gì thần thánh? 71**](#_4anzqyu)

[**5.4 Unity Dependency Injection giúp ích gì cho chúng ta? 72**](#_2pta16n)

[**5.5 Tại sao CSDL lại không có liên kết? 73**](#_14ykbeg)

[**5.6 Tìm kiếm dữ liệu mẫu 73**](#_3oy7u29)

[**Chương 6 Kết luận và hướng phát triển 76**](#_243i4a2)

[**6.1 Kết luận 76**](#_j8sehv)

[**6.2 Hướng phát triển 76**](#_338fx5o)

[**Tài liệu tham khảo 77**](#_1idq7dh)

# Danh mục hình vẽ

[Hình 1 Biểu đồ khảo sát thực trạng thái độ của sinh viên Bách Khoa với hệ thống thông tin sinh viên hiện tại (số người khảo sát: 44 người) 5](#_z337ya)

[Hình 2 Biểu đồ khảo sát mức độ hài long của sinh viên về trang đăng ký tín chỉ (số người khảo sát: 44 người) 6](#_3j2qqm3)

[Hình 3 Biểu đồ khảo sát ý kiến sinh viên về nâng cấp hệ thống (số người khảo sát: 44 người) 6](#_1y810tw)

[Hình 4 Biểu đồ use case tổng quan 7](#_1ci93xb)

[Hình 5 Biểu đồ use case phân rã quản lý thông tin 9](#_1pxezwc)

[Hình 6 Biểu đồ use case phân rã cho quản lý học phần 10](#_49x2ik5)

[Hình 7 Biểu đồ use case phân rã cho quản lý lớp học 11](#_2p2csry)

[Hình 8 Biểu đồ quy trình nghiệp vụ thêm/sửa thông tin sinh viên 12](#_3o7alnk)

[Hình 9 Biểu đồ quy trình nghiệp vụ đăng ký học phần 13](#_23ckvvd)

[Hình 10 Biểu đồ quy trình nghiệp vụ đăng ký lớp học 14](#_ihv636)

[Hình 11 Biểu đồ hoạt động thêm sinh viên 18](#_vx1227)

[Hình 12 Biểu đồ hoạt động tìm sinh viên 21](#_2u6wntf)

[Hình 13 Biểu đồ hoạt động sửa sinh viên 24](#_nmf14n)

[Hình 14 Biểu đồ hoạt động tìm học phần 27](#_2lwamvv)

[Hình 15 Biểu đồ hoạt động đăng ký học phần 31](#_4k668n3)

[Hình 16 Biểu đồ hoạt động đăng ký lớp học 34](#_2dlolyb)

[Hình 17 Web server 40](#_kgcv8k)

[Hình 18 Web service 40](#_34g0dwd)

[Hình 19 Statesless 42](#_1jlao46)

[Hình 20 Web API 43](#_2iq8gzs)

[Hình 21 Repository Pattern 44](#_3hv69ve)

[Hình 22 Unit of work 45](#_4h042r0)

[Hình 23 Chức năng của Unit of Work 46](#_2w5ecyt)

[Hình 24 Mô hình Dependency Injection 47](#_3vac5uf)

[Hình 25 Unity ASPNet Web API và Unity Container 48](#_2afmg28)

[Hình 26 Package diagram 51](#_1302m92)

[Hình 27 Biểu đồ chi tiết package Controller, Model và Common 52](#_2250f4o)

[Hình 28 Biểu đồ chi tiết package Service và Repository 53](#_haapch)

[Hình 29 Form đăng nhập 54](#_40ew0vw)

[Hình 30 Form thêm thông tin sinh viên 55](#_2fk6b3p)

[Hình 31 Biểu đồ lớp của lớp Student 55](#_3ep43zb)

[Hình 32 Biểu đồ lớp Student Controller 56](#_1tuee74)

[Hình 33 Biểu đồ lớp StudentService 56](#_4du1wux)

[Hình 34 Biểu đồ lớp Student Repository 56](#_2szc72q)

[Hình 35 Biểu đồ tuần tự tìm sinh viên 56](#_184mhaj)

[Hình 36 Biểu đồ tuần tự thêm học sinh 57](#_3s49zyc)

[Hình 37 Biểu đồ tuần tự Xóa học sinh 57](#_279ka65)

[Hình 38 Biểu đồ Cơ sở dữ liệu 58](#_36ei31r)

[Hình 39 Kiến trúc cơ bản của Web API 66](#_3bj1y38)

# Danh mục bảng

[Bảng 1 Bảng mô tả quyền hạn của Quản lý trong các use-case 8](#_3whwml4)

[Bảng 2 Bảng mô tả quyền hạn của Giảng viên trong các use-case 8](#_2bn6wsx)

[Bảng 3 Bảng mô tả quyền hạn của Sinh viên trong các use-case 8](#_qsh70q)

[Bảng 4 Đặc tả use case thêm sinh viên 15](#_41mghml)

[Bảng 5 Mô tả dữ liệu đầu vào Thêm sinh viên 16](#_2grqrue)

[Bảng 6 Đặc tả use case Tìm sinh viên 19](#_1v1yuxt)

[Bảng 7 Mô tả dữ liệu đầu vào Tìm sinh viên 20](#_4f1mdlm)

[Bảng 8 Đặc tả use case Sửa thông tin sinh viên 22](#_3tbugp1)

[Bảng 9 Mô tả dữ liệu đầu vào Sửa thông tin sinh viên 23](#_28h4qwu)

[Bảng 10 Đặc tả use case Tim học phần 25](#_1mrcu09)

[Bảng 11 Mô tả dữ liệu Tim học phần 26](#_46r0co2)

[Bảng 12 Đặc tả use case đăng ký học phần 28](#_3l18frh)

[Bảng 13 Mô tả dữ liệu Đăng ký học phần 29](#_206ipza)

[Bảng 14 Đặc tả use case đăng ký lớp học 32](#_1egqt2p)

[Bảng 15 Mô tả dữ liệu Tìm lớp học 33](#_3ygebqi)

[Bảng 16 Danh sách yêu cầu phi chức năng 35](#_3cqmetx)

[Bảng 17 Mô tả Kiến trúc chương trình 49](#_48pi1tg)

[Bảng 18 Mô tả CSDL bảng Student 58](#_1ljsd9k)

[Bảng 19 Mô tả CSDL bảng TheClass 59](#_45jfvxd)

[Bảng 20 Mô tả CSDL bảng Account 59](#_2koq656)

[Bảng 21 Thư viện và công cụ sử dụng 60](#_1yyy98l)

[Bảng 22 Thống kê kết quả đạt được 61](#_2y3w247)

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| API | Application Programming Interface  Giao diện lập trình ứng dụng |
| **EUD** | End-User Development  Phát triển ứng dụng người dùng cuối |
| **GWT** | Google Web Toolkit  Công cụ lập trình Javascript bằng Java của Google |
| **HTML** | HyperText Markup Language  Ngôn ngữ đánh dấu siêu văn bản |
| **CNTT** | Công nghệ thông tin |
| **ĐATN** | Đồ án tốt nghiệp |
| **SV** | Sinh viên |

# Danh mục thuật ngữ

|  |  |
| --- | --- |
| Browser | Trình duyệt |
| **Cache memory** | Bộ nhớ đệm |
| **E-commerce** | Thương mại điện tử |
| **Bloatware** | Ứng dụng nhà sản xuất tích hợp vào thiết bị |
| **Interpreter** | Trình thông dịch |
| **Compiler** | Trình biên dịch |

# Giới thiệu đề tài

## Đặt vấn đề

Thưa thầy cô và các bạn sinh viên Bách Khoa thân mến, là một sinh viên năm cuối, em đã trải qua rất nhiều lần đăng ký tín chỉ với những đêm thức trắng để canh cổng thông tin. Sinh viên chúng ta thì đông mà hệ thống nhà trường thì có hạn, dẫn đến quá tải và tắc nghẽn không thể truy cập. Chắc hẳn trong đời sinh viên của mình, bạn nào cũng gặp phải hoàn cảnh đó. Các thầy cô và cán bộ nhà trường cũng trăn trở rất nhiều vấn đề này. Làm thế nào để thiết kế một hệ thống thông tin cho sinh viên thật mượt mà, suôn sẻ. Đã có các giải pháp đưa ra như tách riêng module đăng ký tín chỉ thành một trang riêng, cũng đã giải quyết được phần nào nhưng vẫn bị quá tải. Điều này cũng diễn ra tương tự với hầu hết các trường Đại học tại Việt Nam. Nguyên nhân xuất phát từ việc hệ thống server còn có cấu hình hạn chế, các công nghệ phần mềm sử dụng cũng đã lạc hậu, nhiều hệ thống còn sử dụng các công nghệ đã cũ như mô hình 3 tầng, MVC, webform, … cùng với đó là ý thức chưa tốt của các bạn sinh viên, khi hệ thống bị chậm, lag thì lại mở thêm nhiều tab mới, refresh trang liên tục khiến số lượng request gửi lên server tăng càng nhiều hơn dẫn đến hệ thống càng trở nên quá tải và treo.

Thêm vào đó, sự bùng nổ của các thiết bị di động cũng như sự phát triển mạnh mẽ của ứng dụng desktop (như Appstore của Microsoft từ Windows 8) cũng đặt ra một vấn đề là phát triển các ứng dụng đa nền tảng, chạy trên nhiều thiết bị khác nhau để mang đến trải nghiệm và chất lượng phục vụ tốt nhất cho người dùng.

Trong quá trình thực tập tại MISA, em được biết đến hệ thống QLTH – hệ thống quản lý giáo dục cho hàng ngàn trường học tại Việt Nam, phục vụ cho hàng trăm ngàn học sinh, phụ huynh, giáo viên, cán bộ quản lý mà vẫn hoạt động hoàn toàn mượt mà, trơn tru. Từ đó, em quyết định học hỏi về công nghệ và ứng dụng vào hệ thống quản lý sinh viên, với mong muốn giải quyết phần nào vấn đề mà chính mình đã trải qua.

Việc sử dụng công nghệ Microsoft .NET WEB API sẽ giúp cho hệ thống có hiệu năng cao hơn, hệ thống có thể chia thành nhiều module nhỏ, linh hoạt và giảm tải tốt hơn. Ngoài ra công nghệ này còn khiến cho việc truy cập từ điện thoại và desktop cũng tương tự như việc truy cập bằng website, giúp tạo nên các hệ thống đa nền tảng một cách dễ dàng hơn, chúng ta chỉ cần tạo 1 back-end cho tất cả các thiết bị và thiết kế front-end cho mỗi kiểu thiết bị khác nhau. Ứng dụng vào hệ thống quản lý sinh viên, nó sẽ khiến cho việc xử lý truy cập của hàng chục nghìn sinh viên 1 lúc dễ dàng hơn cũng như thiết kế một hệ thống quản lý sinh viên đa nền tảng, linh hoạt hơn.

## Mục tiêu và phạm vi đề tài

Hiện nay tất cả các trường đều có hệ thống quản lý và thông tin sinh viên trên internet, tuy nhiên, như em đã trình bày ở trên, hầu hết các hệ thống đều đã cũ và thường xuyên bị quá tải khi đến đợt đăng ký tín chỉ, xem điểm. Một trong số đó là trang sis.hust.edu.vn của sinh viên Bách Khoa chúng ta. Các bạn sinh viên than phiền rất nhiều về hệ thống. Cứ mỗi mùa đăng ký tín chỉ đến là chuyện về SIS lại ngập tràn trên mạng. Nhiều bạn đành “sống chung với lũ”, lạc quan biến những khó khan đó thành chuyện cười như “sinh viên dãn dây chằng ngón tay vì bấm F5 nhiều quá, sinh viên bị người yêu bỏ vì 1 tuần liền ôm máy tính không ôm người yêu, sinh viên cơ khí thất học về quê dựng chuồng nuôi bò vì không đăng ký được tín chỉ”. Các thầy cô và cán bộ quản lý hết sức trăn trở, có nhiều giải pháp đề ra để giải quyết vấn đề này, nhằm phục vụ tốt hơn quá trình học tập của các bạn sinh viên.

Khi phát triển Hệ thống Quản lý trường học (QLTH), Công ty MISA cũng gặp phải vấn đề tương tự, khi phải xây dựng một hệ thống phục vụ cho hang chục ngàn trường học, hang tram ngàn học sinh, giáo viên, phụ huynh và cán bộ quản lý đồng thời sử dụng. Để giải quyết vấn đề này, công ty đã áp dụng nhiều công nghệ, giải pháp để đảm bảo chất lượng cũng như hiệu năng trong hệ thống của mình hoạt động trơn tru, mượt mà, an toàn.

Trong quá trình thực tập tại MISA, rất may mắn cho em được tiếp cận với các công nghệ của công ty. Một trong số đó là Microsoft .NET Web API – công nghệ mới nhất của Microsoft cho việc thiết kế hệ thống HTTP Service. Từ đó, em nảy ra ý tưởng “Tại sao không mang công nghệ này thử giải quyết vấn đề mà chính những sinh viên như mình gặp phải xem sao”.

Việc sử dụng kiến trúc và công nghệ mà MISA đã áp dụng, tận dụng kinh nghiệm từ những người đi trước sẽ giúp tạo nên một hệ thống có hiệu năng tốt hơn, linh hoạt hơn.

Tuy nhiên, với khả năng và thời gian của mình, em không thể thực hiện được một hệ thống hoàn chỉnh, vậy nên em thực hiện một phần của hệ thống, chủ yếu là kiểm tra xem công nghệ nêu trên có đáp ứng được yêu cầu của bài toán quản lý sinh viên hay không

## Định hướng giải pháp

Bằng những tìm hiểu đã kể trên, em quyết định xây dựng hệ thống quản lý sinh viên, về mặt nghiệp vụ vẫn tham khảo theo nghiệp vụ của Hệ thống Quản lý sinh viên trường Đại học Bách Khoa Hà Nội mà cụ thể là trang sis.hust.edu.vn, hệ thống vẫn chạy trên nền tảng web như hiện tại vì đây là nền tảng thân thuộc và dễ sử dụng cũng như đảm bảo thực hiện đầy đủ các chức năng nghiệp vụ.

Về front-end, hệ thống vẫn sử dụng các công nghệ thông dụng và phổ biến hiện nay như html, css, javascript, ngoài ra sử dụng them các thư viện hỗ trợ khác như Jquery, Jquery UI để giúp việc hoàn thiện giao diện dễ dàng hơn và đơn giản hơn cho việc viết code.

Hệ thống back-end được xây dựng trên nền tảng chính là framework .NET API của Microsoft, sử dụng kiến trúc REST (RESTful API) trong xây dựng API, Enity Framework cho việc đóng gói thực thể đối tượng, viết trên ngôn ngữ C#.

Kết quả đạt được là 1 hệ thống quản lý học tập đơn giản gồm 3 module chính: quản lý thông tin sinh viên, quản lý môn học và quản lý lớp học. Bao gồm các chức năng xem, thêm, sửa, xóa, tìm kiếm cho mỗi module.

## Bố cục đồ án

Chương 1 của đồ án giới thiệu về đề tài. Tại sao em lại chọn đề tài này, nguyên nhân, thực trạng ra sao và hướng giải pháp nào em đưa ra để giải quyết các vấn đề đấy.

Chương 2 là các khảo sát của em về đề tài này, cùng với đó là phân tích các yêu cầu mà đề tài đặt ra như các use-case, quy trình nghiệp vụ, các luồng chính của sự kiện, các yêu cầu chức năng và phi chức năng.

Chương 3 là các công nghệ em sử dụng trong chương trình, bao gồm các công nghệ về front-end (tương tác với người dùng) và back-end (tương tác trong hệ thống).

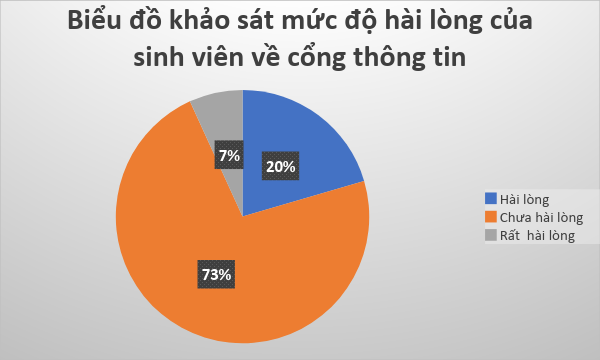
Chương 4 nói về thiết kế và triển khai chương trình. Trong chương này em sẽ nói về kiến trúc của chương trình, trình bày cụ thể về các tầng của kiến trúc, cùng với đó là các thiết kế giao diện, CSDL, và việc triển khai chương trình.

Chương 5 là đóng góp và giải pháp, em sẽ nêu cụ thể tác dụng của các công nghệ mà em sử dụng được áp dụng cho vấn đề nào, giải quyết được những gì, những khó khăn em gặp phải trong quá trính làm đồ án và cách khắc phục.

Chương 6 là đánh giá của em về kết quả đạt được sau quá trình làm đồ án, ưu điểm và nhược điểm cũng như phương hướng phát triển trong tương lai.

# Khảo sát và phân tích yêu cầu

## Khảo sát hiện trạng



Hình 1 Biểu đồ khảo sát thực trạng thái độ của sinh viên Bách Khoa với hệ thống thông tin sinh viên hiện tại (số người khảo sát: 44 người)



Hình 2 Biểu đồ khảo sát mức độ hài lòng của sinh viên về trang đăng ký tín chỉ (số người khảo sát: 44 người)



Hình 3 Biểu đồ khảo sát ý kiến sinh viên về nâng cấp hệ thống (số người khảo sát: 44 người)

Em đã thực hiện khảo sát trên 44 bạn sinh viên, phần lớn trong số họ chưa hài lòng với hệ thống thông tin sinh viên hiện tại và mong muốn có một hệ thống tốt hơn cũng như cải thiện thêm ý thức của các bạn sinh viên khi sử dụng hệ thống như mở quá nhiều tab một lúc, refresh trang liên tục làm tăng khả năng tắc nghẽn server.

Trên cơ sở đó, em thấy việc phát triển một hệ thống mới cho việc quản lý học tập của sinh viên là cần thiết. Hệ thống cần cung cấp các chức năng cho việc đăng kí học tập, quản lý học tập cho cả giảng viên và sinh viên và giải quyết được một số vấn đề về hiệu năng.

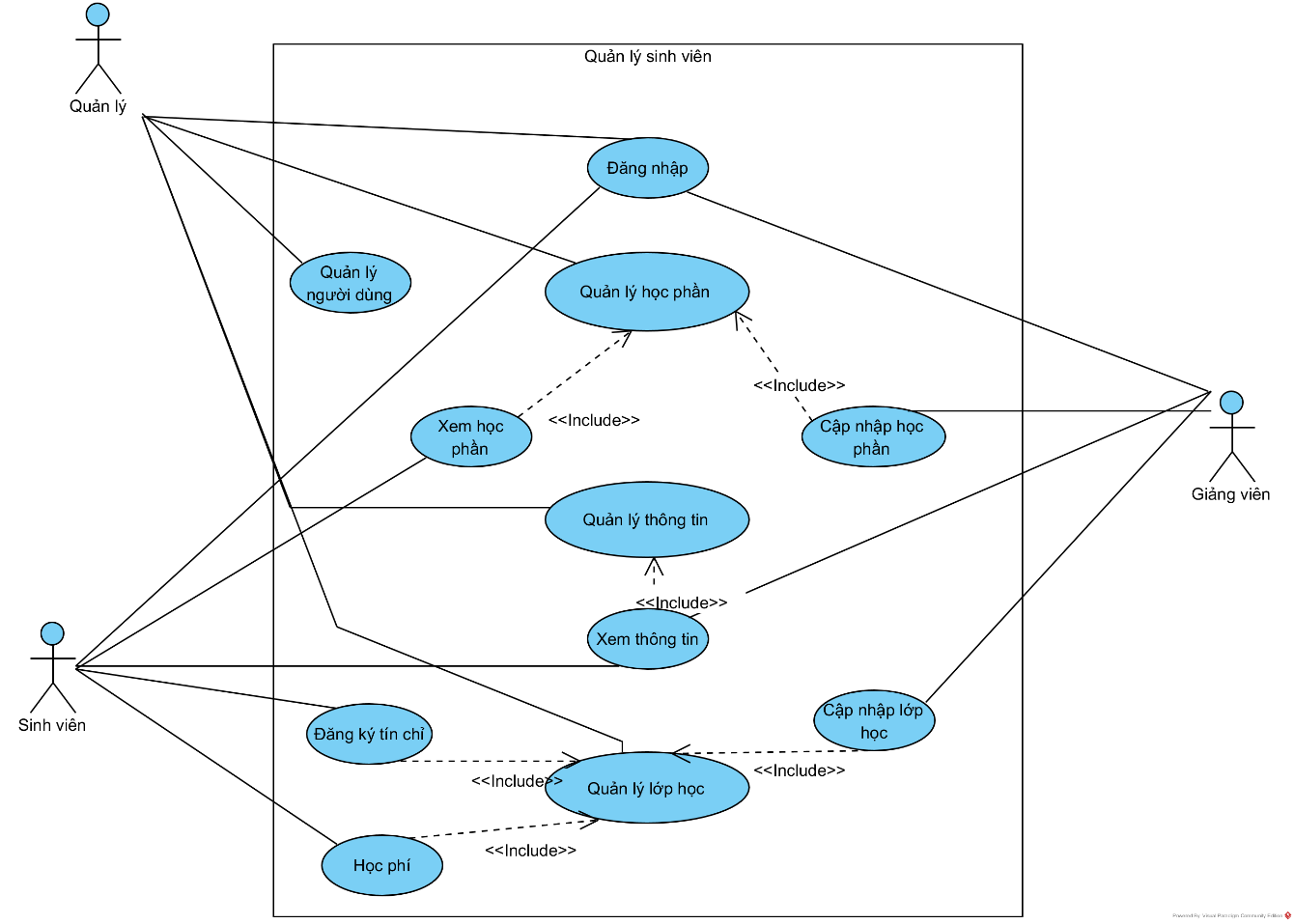
## Tổng quan chức năng

Trong phần này, em xin được phác hoạ các chức năng tổng quan của hệ thống quản lý học tập cho giảng viên và sinh viên thông qua việc xây dựng các biểu đồ trường hợp sử dụng (use-case diagrams).

Hệ thống có 3 tác nhân chính là:

* Quản lý
* Giảng viên
* Sinh viên

### Biểu đồ use case tổng quan



Hình 4 Biểu đồ use case tổng quan

Hệ thống có 3 use-case chính: Quản lý học phần, Quản lý thông tin và Quản lý lớp học. Ngoài ra còn các use-case phụ khác như đăng nhập, xem học phí …

Vì là hệ thống nghiệp vụ nội bộ nên sẽ không có đăng ký tài khoản, toàn bộ việc quản lý tài khoản, phân quyền… được thực hiện bởi người Quản lý thông qua use-case Quản lý người dùng.

Bảng 1 Bảng mô tả quyền hạn của Quản lý trong các use-case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tìm kiếm | Thêm | Sửa | Xóa |
| Module Quản lý học phần | x | x | x | x |
| Module Quản lý thông tin | x | x | x | x |
| Module Quản lý lớp học | x | x | x | x |

Bảng 2 Bảng mô tả quyền hạn của Giảng viên trong các use-case

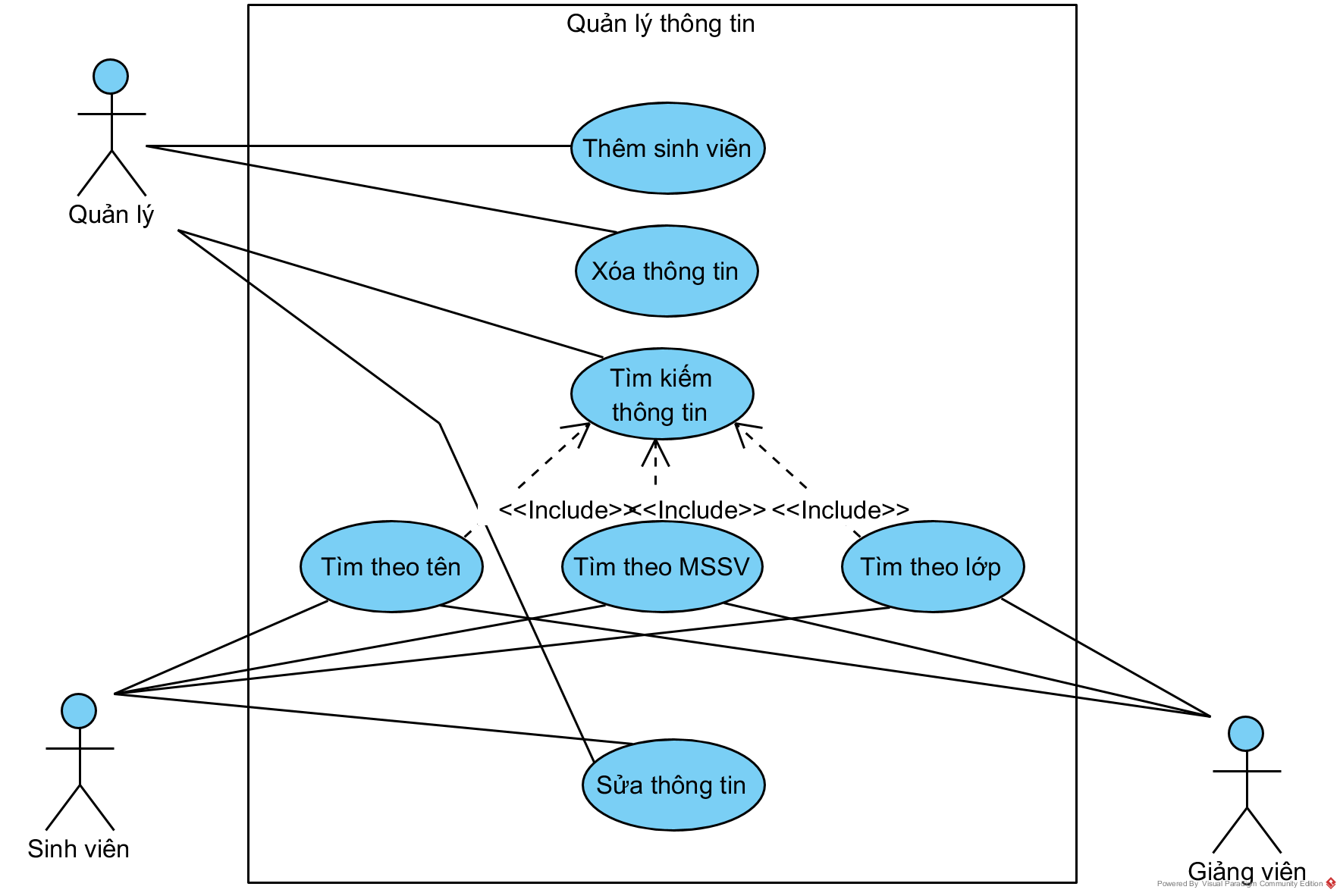
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tìm kiếm | Thêm | Sửa | Xóa |
| Module Quản lý học phần | x | x | x | x |
| Module Quản lý thông tin | x |  |  |  |
| Module Quản lý lớp học | x | x | x | x |

Bảng 3 Bảng mô tả quyền hạn của Sinh viên trong các use-case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tìm kiếm | Thêm | Sửa | Xóa |
| Module Quản lý học phần | x |  |  |  |
| Module Quản lý thông tin | x |  | x |  |
| Module Quản lý lớp học | x |  |  |  |

### Biểu đồ use case phân rã

1. Biểu đồ use case phân rã Quản lý thông tin



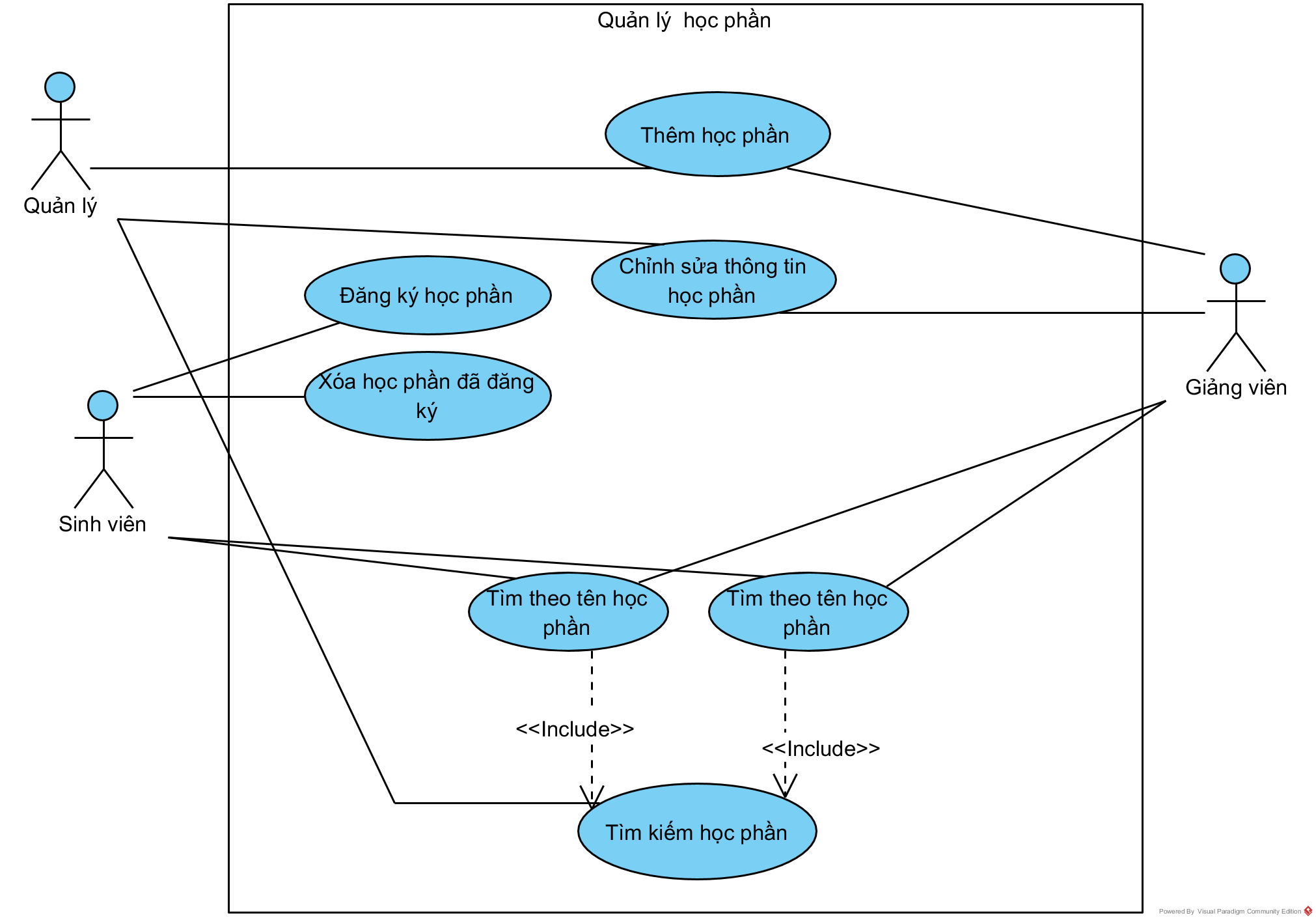
Hình 5 Biểu đồ use case phân rã quản lý thông tin

Bao gồm 3 tác nhân chính: quản lý, giảng viên và sinh viên.

4 use case chính: thêm, tìm kiếm, sửa, xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Use case phân rã cho Quản lý học phần



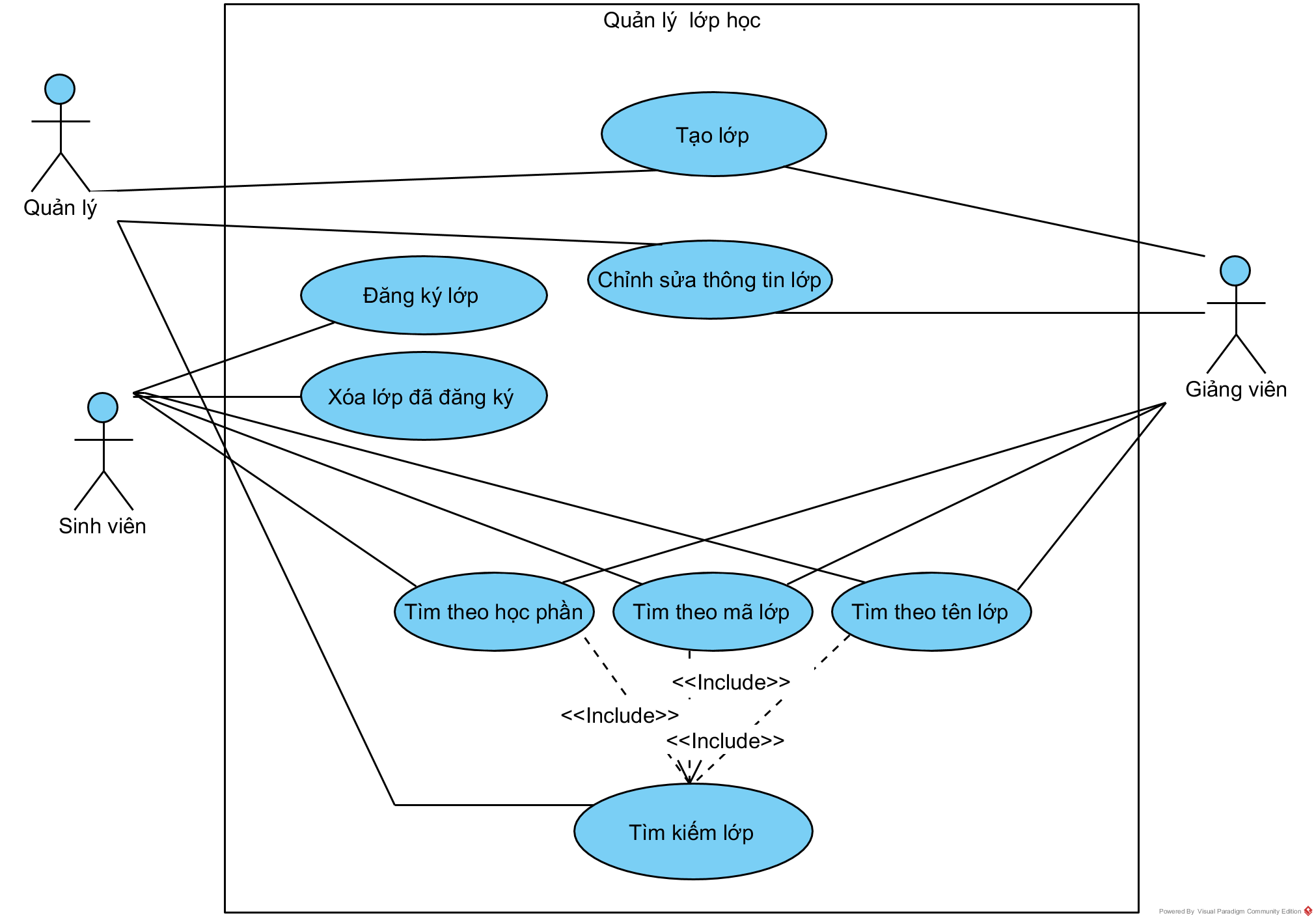
Hình 6 Biểu đồ use case phân rã cho quản lý học phần

Bao gồm 3 tác nhân chính: quản lý, giảng viên và sinh viên.

4 use case chính: thêm, tìm kiếm, sửa, xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

1. Use case phân rã cho Quản lý lớp học



Hình 7 Biểu đồ use case phân rã cho quản lý lớp học

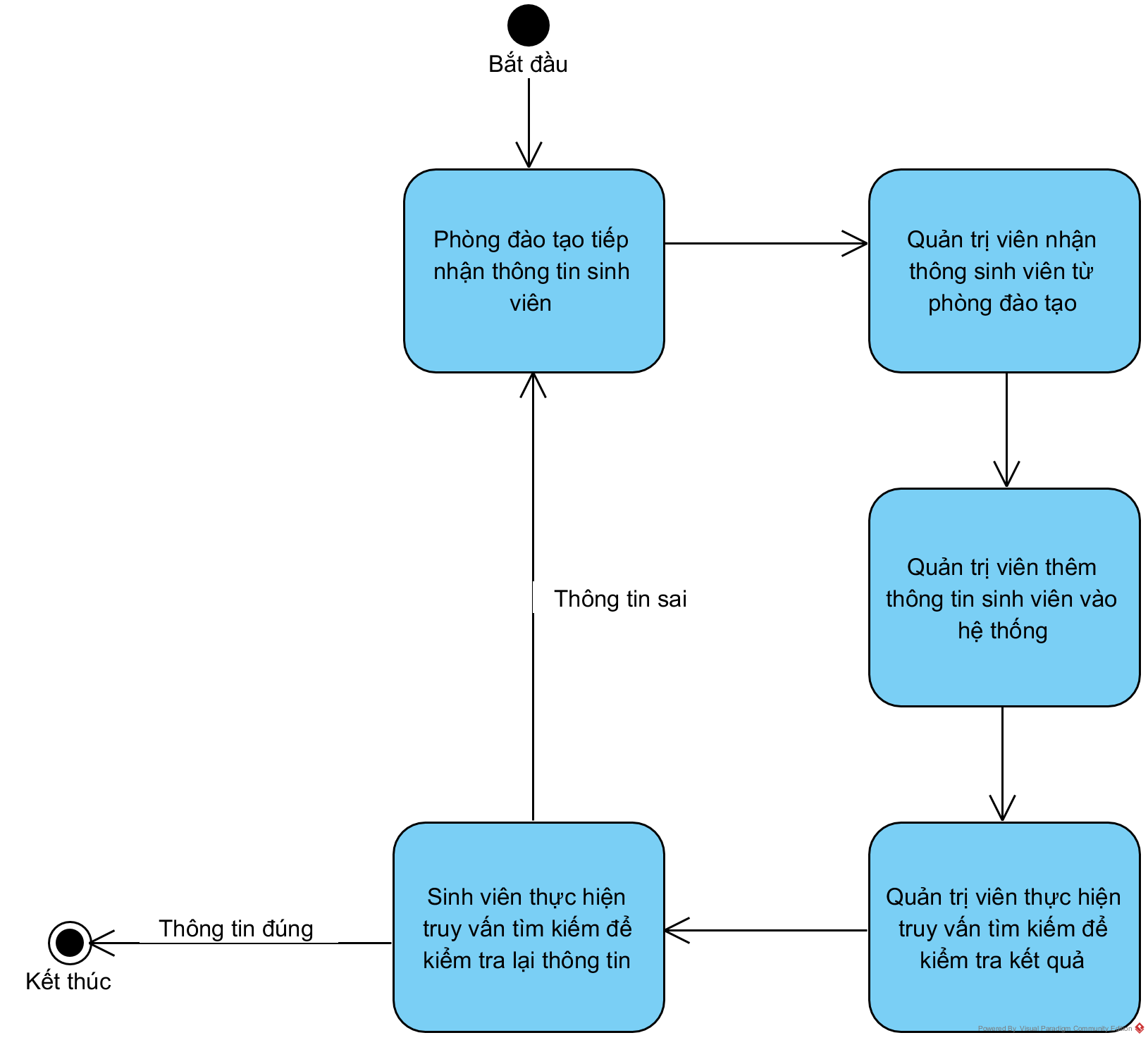
Bao gồm 3 tác nhân chính: quản lý, giảng viên và sinh viên.

4 use case chính: thêm, tìm kiếm, sửa ,xóa.

Các tác nhân được sử dụng các use-case theo quyền hạn đã được liệt kê ở 2.2.1

### Quy trình nghiệp vụ

##### Quy trình nghiệp vụ thêm thông tin sinh viên



Hình 8 Biểu đồ quy trình nghiệp vụ thêm/sửa thông tin sinh viên

Mô tả quy trình:

Bước 1: Phòng đào tạo tiếp nhận thông tin từ sinh viên

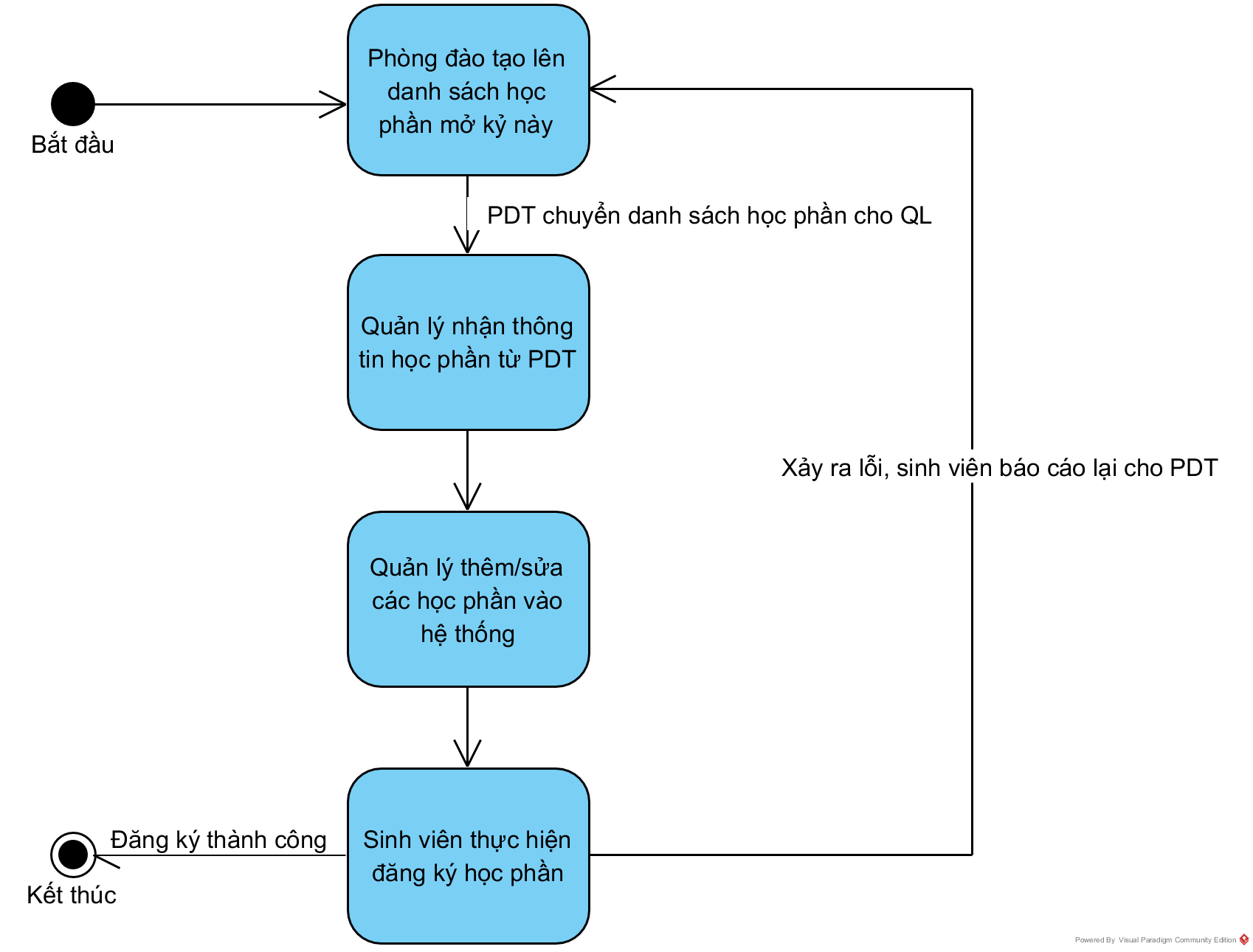
Bước 2: Quản trị viên nhận thông tin sinh viên từ phòng đào tạo

Bước 3: Quản trị viên thêm/sửa thông tin vào hệ thống

Bước 4: Quản trị viên truy vấn để kiểm tra lại thông tin đã thêm/sửa

Bước 5: Sinh viên truy vấn để kiểm tra lại thông tin được yêu cầu, nếu đúng thì kết thúc quy trình, nếu sai thì báo lại thông tin với phòng đào tạo và lặp lại quy trình.

##### Quy trình nghiệp vụ đăng ký học phần



Hình 9 Biểu đồ quy trình nghiệp vụ đăng ký học phần

Mô tả quy trình:

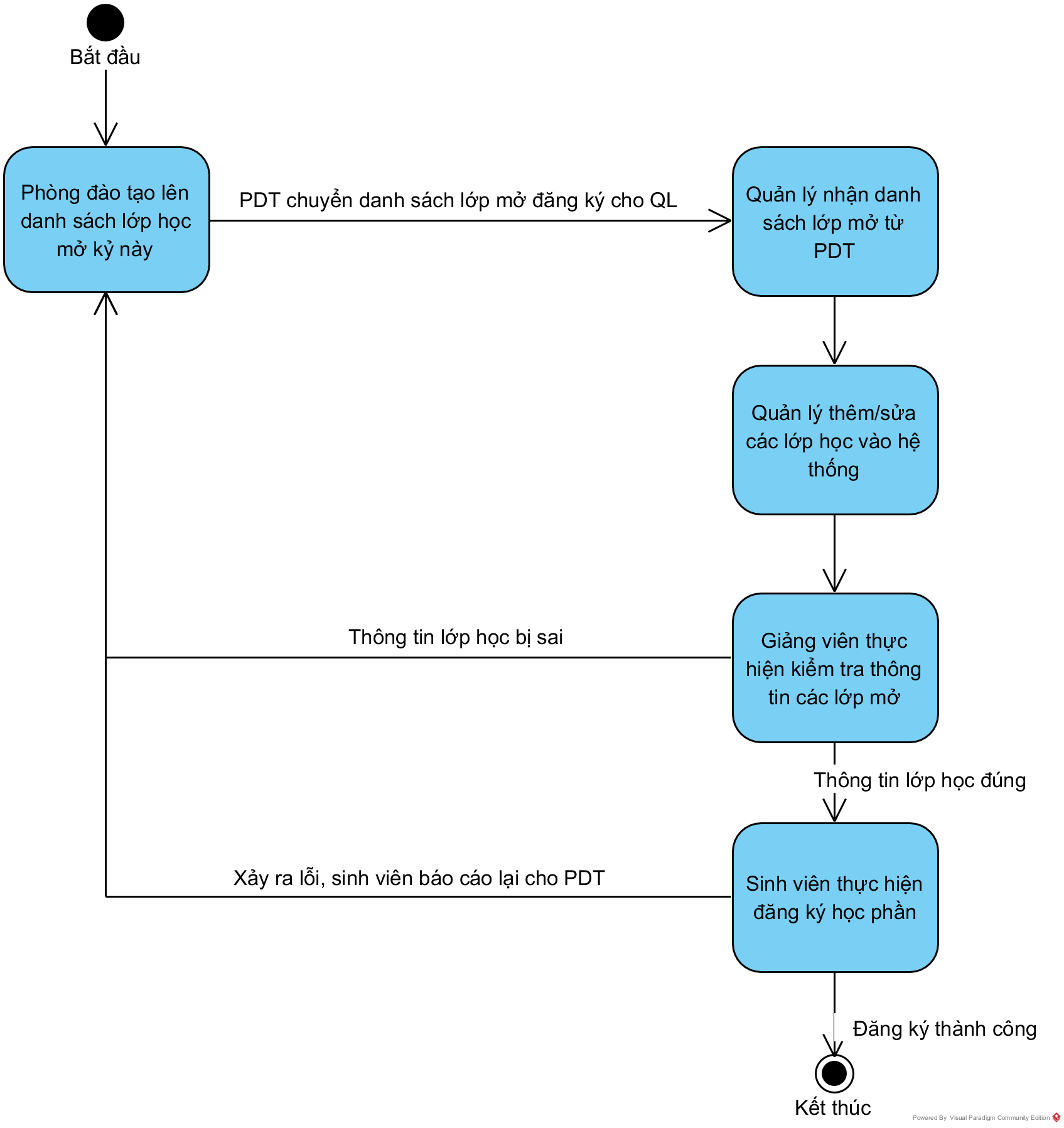
Bước 1: Phòng đào tạo lên danh sách học phần mở đăng ký

Bước 2: Quản trị viên nhận thông tin học phần từ phòng đào tạo

Bước 3: Quản trị viên thêm/sửa thông tin học phần trong hệ thống

Bước 4: Sinh viên thực hiện đăng ký học phần, nếu thành công thì kết thúc quy trình, nếu có lỗi hoặc sai sót thì báo lại thông tin với phòng đào tạo và lặp lại quy trình.

##### Quy trình nghiệp vụ đăng ký lớp học



Hình 10 Biểu đồ quy trình nghiệp vụ đăng ký lớp học

Mô tả quy trình:

Bước 1: Phòng đào tạo lên danh sách lớp học mở đăng ký

Bước 2: Quản trị viên nhận danh sách lớp mở từ phòng đào tạo

Bước 3: Quản trị viên thêm/sửa danh sách lớp học trong hệ thống

Bước 4: Giảng viên thực hiện kiểm tra lại các lớp học đã mở, nếu đúng thì chuyển qua bước tiếp theo, nếu sai thì báo cáo lại phòng đào tạo

Bước 5: Sinh viên thực hiện đăng ký lớp học, nếu đăng ký thành công thì kết thúc quy trình, nếu phát sinh lỗi hoặc sai sót thì báo cáo lại phòng đào tạo,

## Đặc tả chức năng

### Đặc tả use case Thêm sinh viên

##### Mô tả

Quản trị viên thực hiện thêm sinh viên vào hệ thống

##### Luồng sự kiện và phát sinh

Bảng 4 Đặc tả use case thêm sinh viên

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCTSV | **Tên UseCase** | Thêm thông tin sinh viên |
| **Tác nhân** | Quản lý | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Quản trị viên | Vào trang quản lý thông tin sinh viên | |
| 2 | Client | Hiển thị giao diện quản lý thông tin sinh viên | |
| 3 | Quản trị viên | Chọn chức năng thêm sinh viên | |
| 4 | Client | Hiển thị giao diện thêm sinh viên | |
| 5 | Quản trị viên | Nhập thông tin sinh viên | |
| 6 | Client | Kiểm tra thông tin có hợp lệ hay không | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị kết quả truy vấn cho Quản lý | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Hiển thị cảnh báo nếu thông tin nhập vào không hợp lệ | |
| 6b | Client | Hiển thị thông báo nếu thêm không thành công | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của thông tin cá nhân

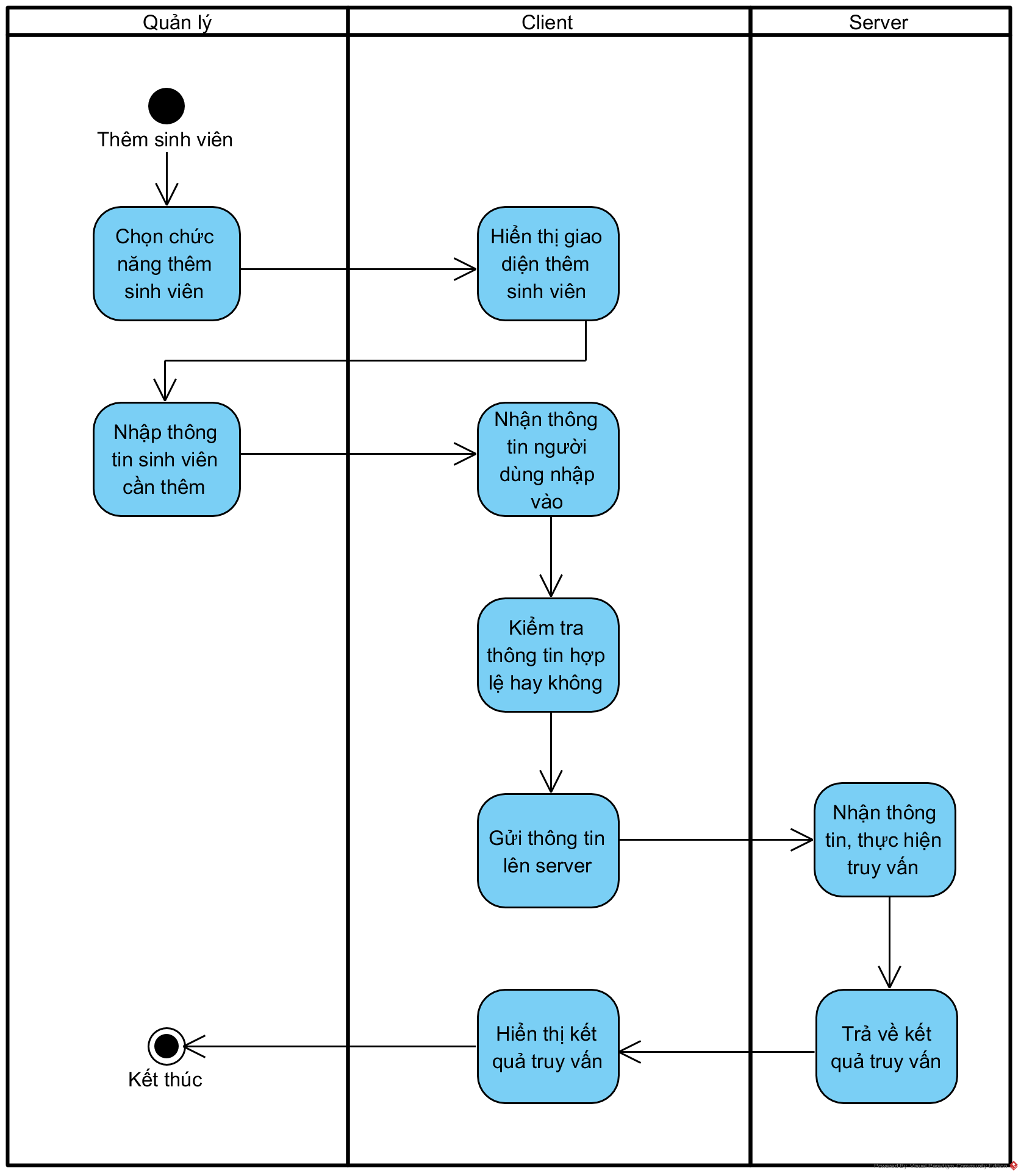
Bảng 5 Mô tả dữ liệu đầu vào Thêm sinh viên

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã sinh viên |  | Có | Là 1 dãy số | 20141234 |
| 2 | Họ và tên |  | Có | Là 1 chuỗi | Trần Đức Nam |
| 3 | Giới Tính | Lựa chọn Nam/Nữ | Có |  | Lựa chọn Nam |
| 4 | Ngày sinh | Lựa chọn Ngày tháng năm | Có | Dạng ngày tháng | 01/05/1996 |
| 5 | Khoa viện | Lựa chọn | Có | Dạng enum | 1 ứng với viện CNTT |
| 6 | Khóa | Lựa chọn | Có | Dạng enum | 60 ứng với K60 |
| 7 | Địa chỉ |  | Không | Là 1 chuỗi | Số 1 Đại Cồ Việt |
| 8 | Trạng thái | Lựa chọn | Có | Dạng enum | 1 ứng với “Đang học” |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động thêm sinh viên



Hình 11 Biểu đồ hoạt động thêm sinh viên

### Đặc tả use case Tìm kiếm sinh viên

##### Mô tả

Thực hiện công việc truy vấn tìm 1 sinh viên hoặc nhiều sinh viên trong hệ thống

##### Luồng sự kiện và phát sinh

Bảng 6 Đặc tả use case Tìm sinh viên

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCTSV2 | **Tên UseCase** | Tìm sinh viên |
| **Tác nhân** | Người dùng (quản lý, giảng viên, sinh viên) | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Người dùng | Vào trang quản lý thông tin sinh viên | |
| 2 | Client | Hiển thị giao diện quản lý thông tin sinh viên | |
| 3 | Người dùng | Điền các ô trong trường tìm kiếm | |
| 4 | Client | Lấy nội dung các ô trong trường tìm kiếm | |
| 5 | Client | Gửi thông tin lên server | |
| 6 | Server | Nhận thông tin, thực hiện truy vấn | |
| 7 | Server | Trả về kết quả truy vấn cho client | |
| 8 | Client | Hiển thị kết quả truy vấn cho người dùng | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 4a | Client | Đóng gói dữ liệu | |
| 5a | Client | Hiển thị icon “Xin hãy đợi” | |
| 5b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 8a | Client | Hiển thị thông báo nếu không lấy được dữ liệu | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của thông tin cá nhân

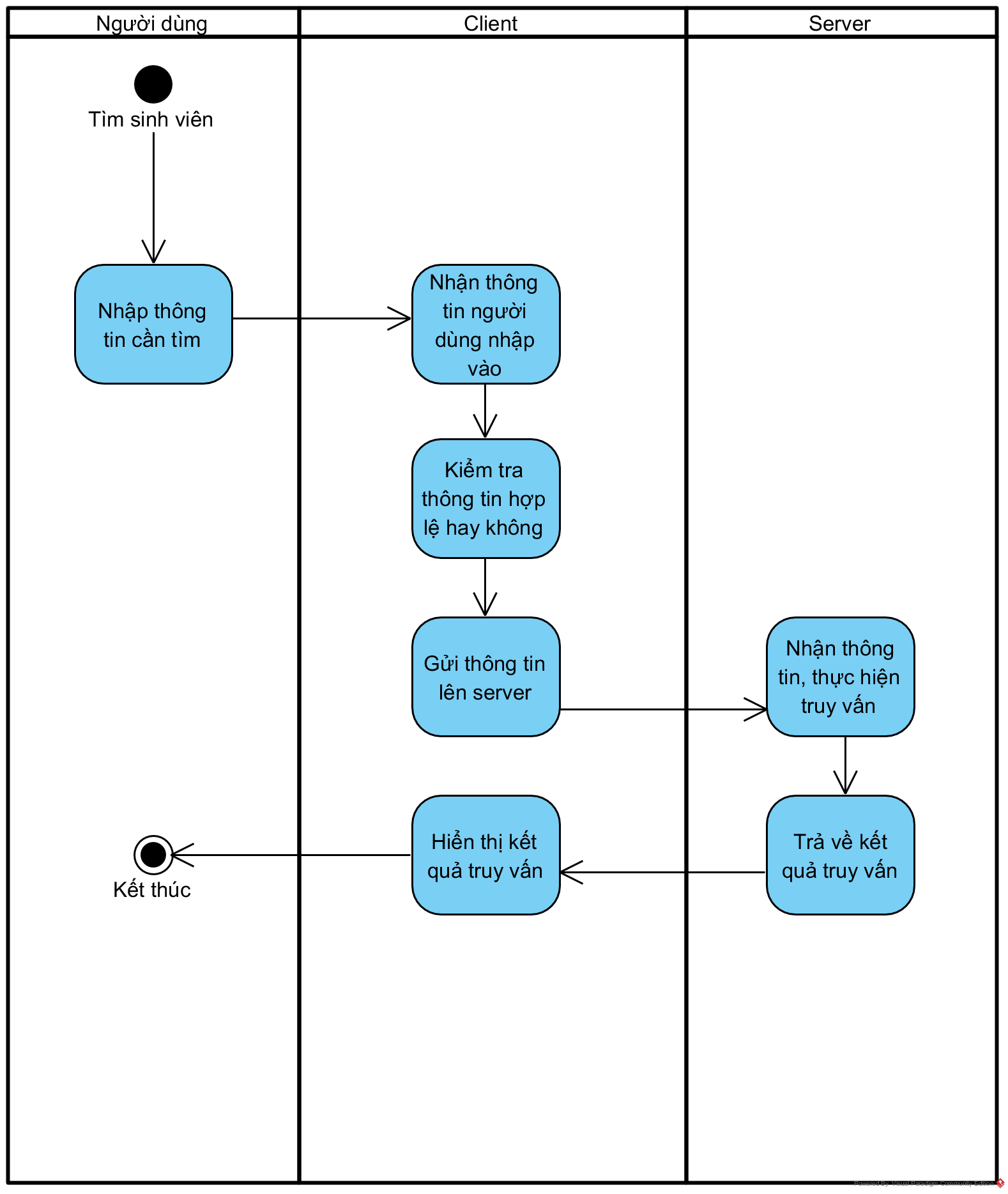
Bảng 7 Mô tả dữ liệu đầu vào Tìm sinh viên

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã sinh viên |  | Có | Là 1 dãy số | 20141234 |
| 2 | Họ và tên |  | Có | Là 1 chuỗi | Trần Đức Nam |
| 3 | Giới Tính | Lựa chọn Nam/Nữ | Có |  | Lựa chọn Nam |
| 4 | Ngày sinh | Lựa chọn Ngày tháng năm | Có | Dạng ngày tháng | 01/05/1996 |
| 5 | Khoa viện | Lựa chọn | Có | Dạng enum | 1 ứng với viện CNTT |
| 6 | Khóa | Lựa chọn | Có | Dạng enum | 60 ứng với K60 |
| 7 | Địa chỉ |  | Không | Là 1 chuỗi | Số 1 Đại Cồ Việt |
| 8 | Trạng thái | Lựa chọn | Có | Dạng enum | 1 ứng với “Đang học” |
| 9 | Yêu cầu lọc dữ liệu | Ứng với mỗi trường dữ liệu sẽ có 1 biến ứng với yêu cầu lọc dữ liệu | Có | Dạng enum | ‘\* ứng với chứa  = ứng với bằng  + ứng với bắt đầu bằng  ‘- ứng với kết thúc bằng  ! ứng với không chứa |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động tìm sinh viên



Hình 12 Biểu đồ hoạt động tìm sinh viên

### Đặc tả use case Sửa thông tin sinh viên

##### Mô tả

Thực hiện công việc sửa thông tin của 1 sinh viên trong hệ thống

##### Luồng sự kiện và phát sinh

Bảng 8 Đặc tả use case Sửa thông tin sinh viên

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCSTT | **Tên UseCase** | Sửa thông tin sinh viên |
| **Tác nhân** | Quản lý | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Quản lý | Vào trang quản lý thông tin sinh viên | |
| 2 | Client | Hiển thị giao diện quản lý thông tin sinh viên | |
| 3 | Quản lý | Chọn giao diện Sửa thông tin | |
| 4 | Client | Hiển thị giao diện Sửa thông tin | |
| 5 | Quản lý | Nhập các thông tin cần sửa | |
| 6 | Client | Lấy nội dung các trường cần sửa | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị kết quả truy vấn cho Quản lý | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị icon “Xin hãy đợi” | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của thông tin cá nhân

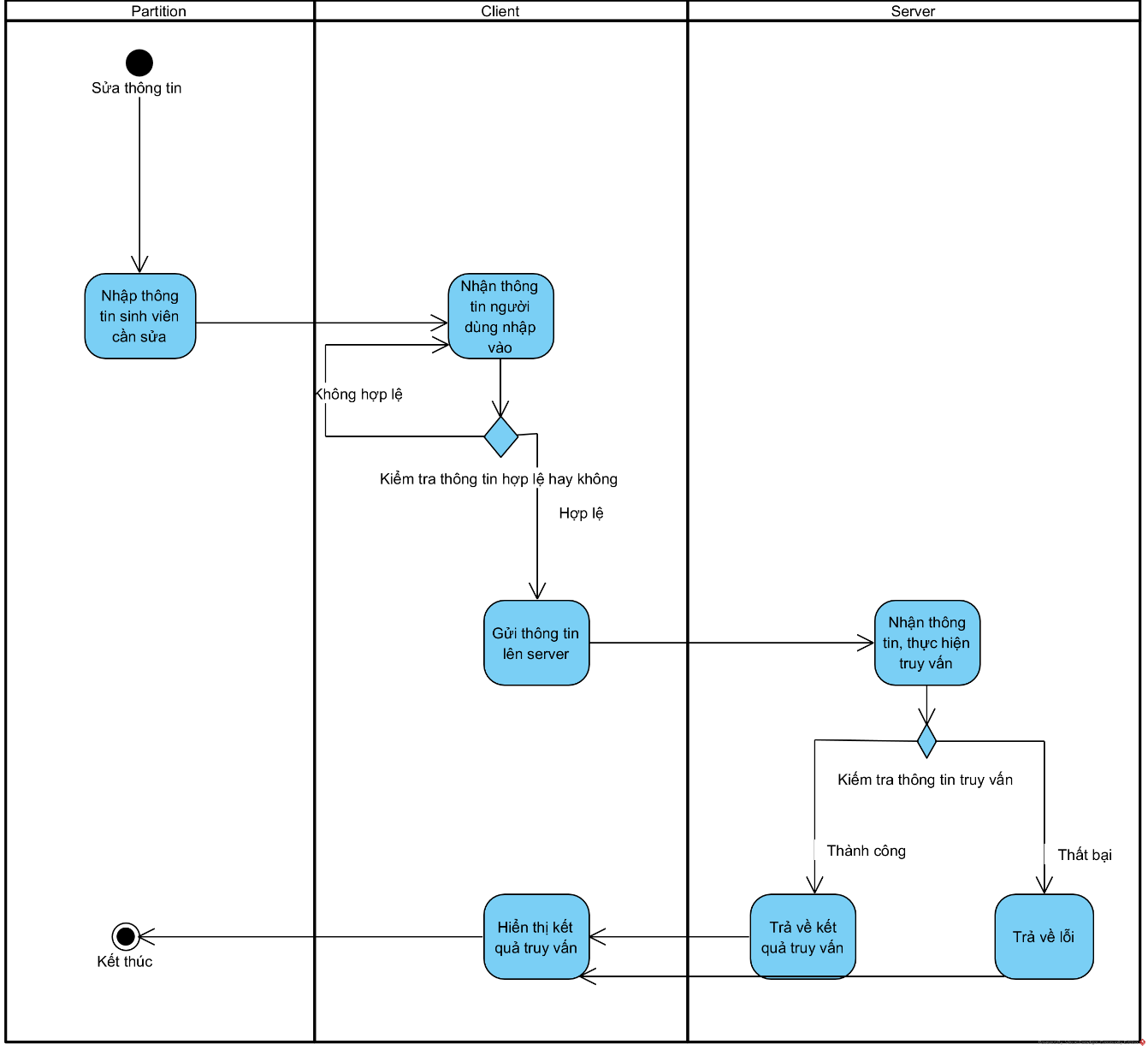
Bảng 9 Mô tả dữ liệu đầu vào Sửa thông tin sinh viên

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã sinh viên |  | Không | Là 1 dãy số | 20141234 |
| 2 | Họ và tên |  | Không | Là 1 chuỗi | Trần Đức Nam |
| 3 | Giới Tính | Lựa chọn Nam/Nữ | Không |  | Lựa chọn Nam |
| 4 | Ngày sinh | Lựa chọn Ngày tháng năm | Không | Dạng ngày tháng | 01/05/1996 |
| 5 | Khoa viện | Lựa chọn | Không | Dạng enum | 1 ứng với viện CNTT |
| 6 | Khóa | Lựa chọn | Không | Dạng enum | 60 ứng với K60 |
| 7 | Địa chỉ |  | Không | Là 1 chuỗi | Số 1 Đại Cồ Việt |
| 8 | Trạng thái | Lựa chọn | Không | Dạng enum | 1 ứng với “Đang học” |

##### Yêu cầu chức năng

Hệ thống hoạt động chính xác, có cảnh báo khi không đạt yêu cầu, hiển thị cảnh báo khi sinh ra lỗi.

##### Mô tả biểu đồ hoạt động sửa thông tin sinh viên



Hình 13 Biểu đồ hoạt động sửa sinh viên

### Đặc tả use case Tìm học phần

##### Mô tả

Sinh viên thực hiện tìm học phần trong trang đăng ký học phần

Sinh viên nhập các thông tin cần tìm kiếm như mã học phần, tên học phần, số tín chỉ… hệ thống sẽ trả về danh sách các học phần phù hợp với dữ liệu tìm được

##### Luồng sự kiện và phát sinh

Bảng 10 Đặc tả use case Tim học phần

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCDKHP | **Tên UseCase** | Tim học phần |
| **Tác nhân** | Sinh viên | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Sinh viên | Vào trang quản lý học phần | |
| 2 | Client | Hiển thị giao diện trang đăng ký | |
| 5 | Sinh viên | Nhập các thông tin học phần cần tìm | |
| 6 | Client | Lấy nội dung các trường cần tìm | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị danh sách học phần | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị icon “Xin hãy đợi” | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của trường học phần

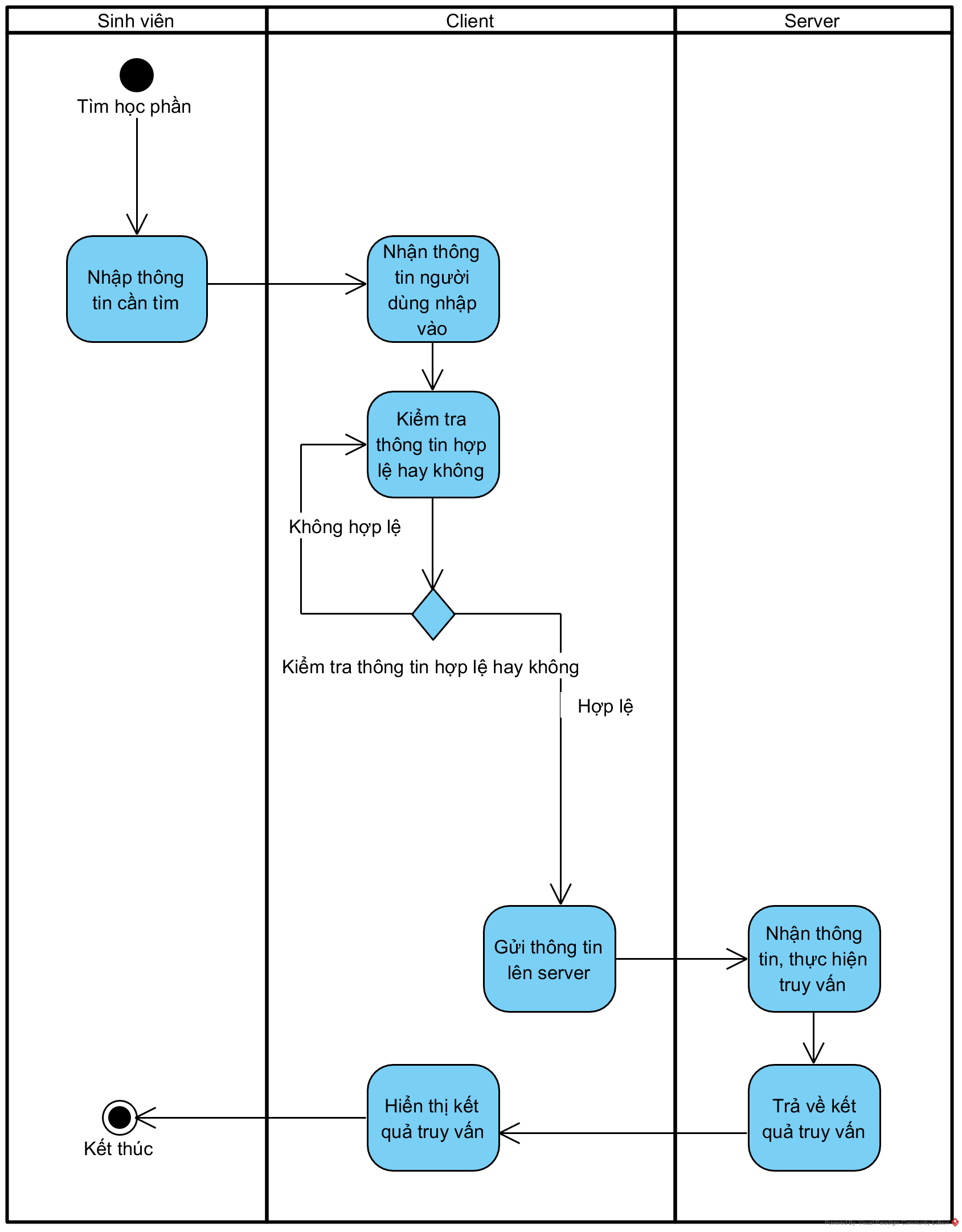
Bảng 11 Mô tả dữ liệu Tim học phần

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã học phần |  | Không | Là 1 chuỗi | MI1110 |
| 2 | Tên học phần |  | Không | Là 1 chuỗi | Giải tích I |
| 3 | Khoa viện | Lựa chọn | Không | Kiểu enum | 1 ứng với viện CNTT |
| 4 | Số tín chỉ |  | Không | Là 1 số | 3 |
| 5 | Số tín chỉ học phí |  | Không | Là 1 số | 4 |
| 6 | Mô tả |  | Không | Là 1 chuỗi | “Mở nửa kỳ” |
| 7 | Trọng số |  | Không | Là 1 số thập phân | 0.7 |
| 8 | Điều kiện tìm kiếm | Mỗi trường có 1 điều kiện tìm kiếm | Có | Kiểu enum | ‘\* ứng với chứa  = ứng với bằng  + ứng với bắt đầu bằng  ‘- ứng với kết thúc bằng  ! ứng với không chứa |

##### Yêu cầu chức năng

Đây là một trong những phần dễ gây tắc nghẽn nhất hệ thống, cần thực hiện với hiệu năng tốt, tốc độ đạt yêu cầu, đảm bảo chính xác và an toàn cho hệ thống

##### Mô tả biểu đồ hoạt động tìm học phần



Hình 14 Biểu đồ hoạt động tìm học phần

### Đặc tả use case đăng ký học phần

##### Mô tả

Sinh viên thực hiện đăng ký học phần trong trang đăng ký

Sinh viên thực hiện tìm học phần, chọn học phần và gửi đăng ký

##### Luồng sự kiện và phát sinh

Bảng 12 Đặc tả use case đăng ký học phần

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCDTHP | **Tên UseCase** | Tìm học phần |
| **Tác nhân** | Sinh viên | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1 | Sinh viên | Vào trang quản lý học phần | |
| 2 | Client | Hiển thị giao diện trang đăng ký | |
| 3 | Sinh viên | Chọn giao diện Đăng ký học phần | |
| 4 | Client | Hiển thị giao diện Đăng ký học phần | |
| 5 | Sinh viên | Nhập các thông tin học phần cần tìm | |
| 6 | Client | Lấy nội dung các trường cần tìm | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị danh sách học phần | |
| 11 | Sinh viên | Chọn học phần đăng ký | |
| 12 | Client | Gửi danh sách học phần đăng ký lên server | |
| 13 | Server | Thực hiện truy vấn và trả kết quả về | |
| 14 | Client | Hiển thị kết quả đăng ký | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị icon “Xin hãy đợi” | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| 12a | Client | Lấy mã ID của học phần | |
| 12b | Client | Gửi thông tin học phần lên server | |
| 13a | Server | Kiểm tra tính hợp lệ của thông tin | |
| 13b | Server | Trả về lỗi nếu truy vấn không thành công | |
| 14a | Client | Hiển thị lỗi nếu đăng ký không thành công | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của trường học phần

Bảng 13 Mô tả dữ liệu Đăng ký học phần

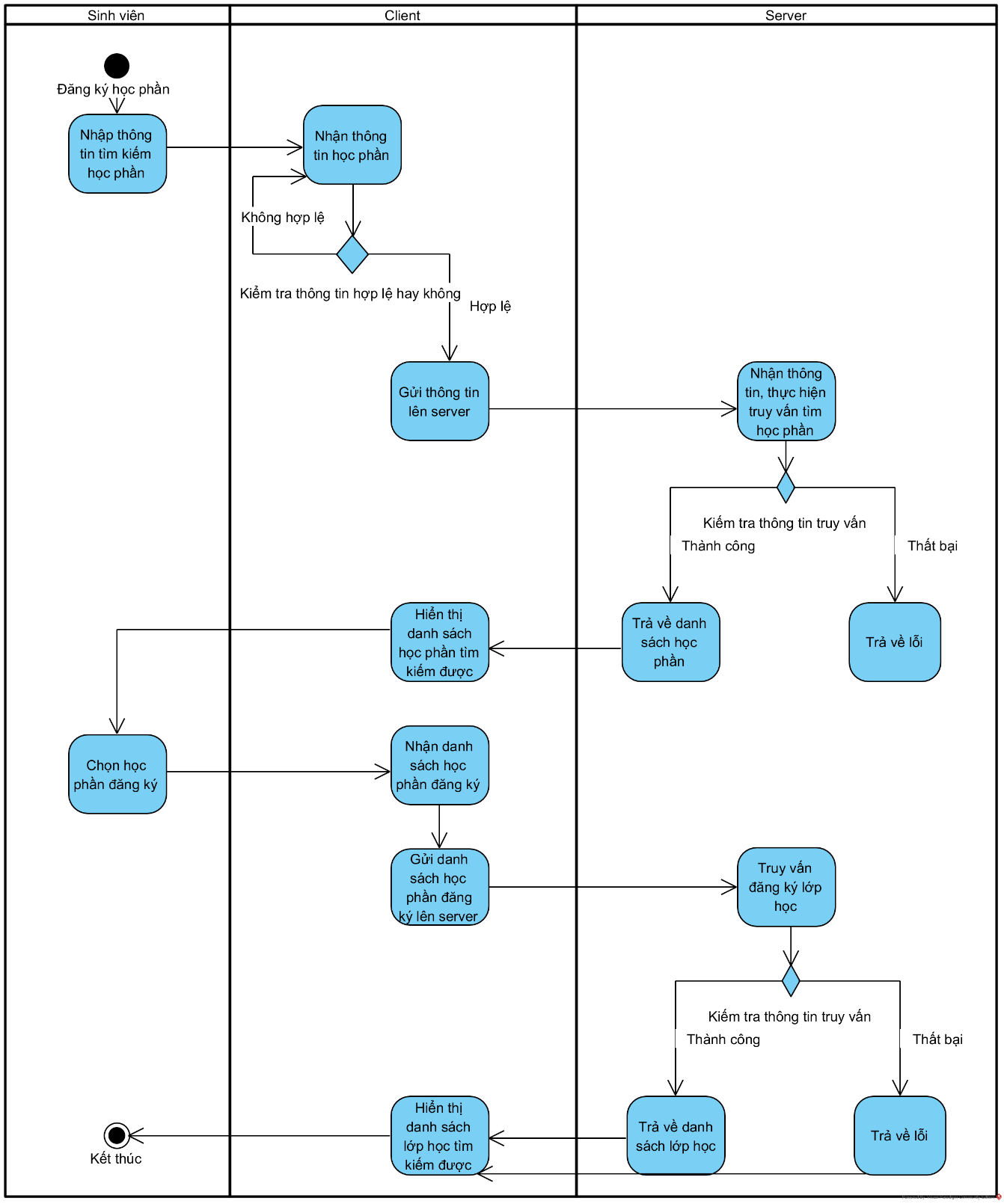
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã học phần |  | Không | Là 1 chuỗi | MI1110 |
| 2 | Tên học phần |  | Không | Là 1 chuỗi | Giải tích I |
| 3 | Khoa viện | Lựa chọn | Không | Kiểu enum | 1 ứng với viện CNTT |
| 4 | Số tín chỉ |  | Không | Là 1 số | 3 |
| 5 | Số tín chỉ học phí |  | Không | Là 1 số | 4 |
| 6 | Mô tả |  | Không | Là 1 chuỗi | “Mở nửa kỳ” |
| 7 | Trọng số |  | Không | Là 1 số thập phân | 0.7 |

##### Yêu cầu chức năng

Bước 1 hiển thị kết quả tìm kiếm, sau khi thực hiện tìm kiếm sẽ cho chọn học phần để đăng ký.

Hệ thống hoạt động chính xác, bảo đảm toàn vẹn dữ liệu và hiệu năng.

##### Mô tả biểu đồ hoạt động đăng ký học phần



Hình 15 Biểu đồ hoạt động đăng ký học phần

### Đặc tả use case đăng ký lớp học

##### Mô tả

Đây là phần hay bị treo nhất trong hệ thống, sinh viên thực hiện đăng ký lớp học cho kỳ học sắp tới

##### Luồng sự kiện và phát sinh

Bảng 14 Đặc tả use case đăng ký lớp học

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã UseCase** | UCDKL | **Tên UseCase** | Đăng ký lớp học |
| **Tác nhân** | Sinh viên | | |
| **Điều kiện trước** | Đăng nhập | | |
| **Luồng thực thi chính** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 3 | Sinh viên | Chọn giao diện Đăng ký lớp học | |
| 4 | Client | Hiển thị giao diện Đăng ký lớp học | |
| 5 | Sinh viên | Nhập các thông tin học phần cần tìm | |
| 6 | Client | Lấy nội dung các trường cần tìm | |
| 7 | Client | Gửi thông tin lên server | |
| 8 | Server | Nhận thông tin, thực hiện truy vấn | |
| 9 | Server | Trả về kết quả truy vấn cho client | |
| 10 | Client | Hiển thị danh sách Lớp học | |
| 11 | Sinh viên | Chọn lớp học đăng ký | |
| 12 | Client | Gửi danh sách lớp học đăng ký lên server | |
| 13 | Server | Thực hiện truy vấn và trả kết quả về | |
| 14 | Client | Hiển thị kết quả đăng ký | |
| **Luồng thực thi mở rộng** | | | |
| **No.** | **Thực hiện** | **Hành động** | |
| 1a | Server | Kiểm tra người dùng đã đăng nhập trước đó chưa | |
| 1b | Client | Hiển thị thông báo nếu truy cập của người dùng không hợp lệ | |
| 6a | Client | Kiểm tra nội dung các trường cần sửa, kiểm tra tính hợp lệ của dữ liệu | |
| 6b | Client | Đóng gói dữ liệu | |
| 7a | Client | Hiển thị icon “Xin hãy đợi” | |
| 7b | Server | Kiểm tra thông tin, kiểm tra SQL Injection | |
| 10a | Client | Hiển thị thông báo nếu không Thực thi được truy vấn | |
| 12a | Client | Lấy mã ID của học phần | |
| 12b | Client | Gửi thông tin lớp học lên server | |
| 13a | Server | Kiểm tra tính hợp lệ của thông tin | |
| 13b | Server | Trả về lỗi nếu truy vấn không thành công | |
| 14a | Client | Hiển thị lỗi nếu đăng ký không thành công | |
| **Điều kiện sau** | Không | | |

##### Mô tả dữ liệu đầu vào của trường tìm kiếm lớp học

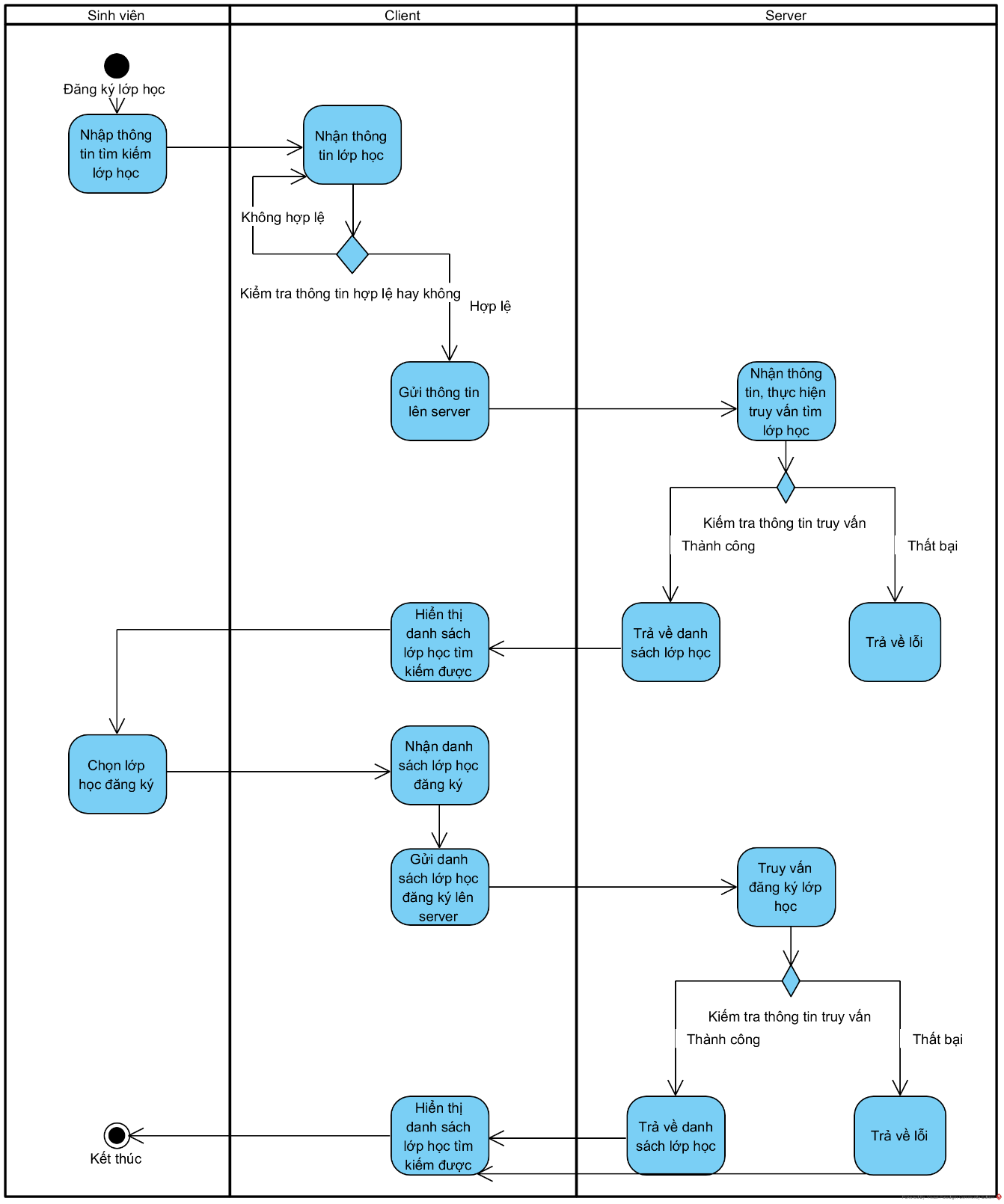
Bảng 15 Mô tả dữ liệu Tìm lớp học

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường Dữ Liệu | Mô Tả | Bắt Buộc | Điều Kiện Hợp Lệ | Ví Dụ |
| 1 | Mã lớp |  | Không | Là 1 chuỗi | 94536 |
| 2 | Mã học phần |  | Không | Là 1 chuỗi | MI1110 |
| 3 | Tên học phần |  | Không | Là 1 chuỗi | Giải tích I |

##### Yêu cầu chức năng

Hệ thống xử lý nhanh, kịp thời, chính xác.

##### Mô tả biểu đồ hoạt động đăng ký lớp học



Hình 16 Biểu đồ hoạt động đăng ký lớp học

## Yêu cầu phi chức năng

Hệ thống được xây dựng cần phải đáp ứng được các yêu cầu sau:

Bảng 16 Danh sách yêu cầu phi chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Yêu cầu | Phân loại yêu cầu | Cụ thể |
| 1 | Hiệu năng hoạt động | Yêu cầu về thời gian | Thời gian truy cập và gửi dữ liệu tới người sử dụng hệ thống cần phải diễn ra một cách nhanh chóng.  Đỗ trễ của hệ thống không quá 5s  Khi trễ quá 5s, hiển thị thông báo lỗi |
|  |  | Tài nguyên sử dụng | Hệ thống sẽ sử dụng các phần cứng khác nhau khi được khởi chạy trên các thiết bị khác nhau.  Có thể triển khai được trên cả Web, Desktop và Moblie |
|  |  | Công suất tối đa | Có thể phục vụ được truy cập của 5000 sinh viên 1 lúc |
| 2 | Tương thích | Cùng tồn tại | Hệ thống khi được khởi chạy trên nhiều thiết bị khác nhau cần phải đảm bảo tất cả đều khởi chạy bình thường, không quan tâm tới ứng dụng được khởi chạy trên môi trường nào |
|  |  | Tương tác liên thông | Hệ thống cần đảm bảo các thông tin được cập nhật liên tục lên cơ sở dữ liệu chung của ứng dụng. Tất cả các dữ liệu được gửi tới các thiết bị có trong hệ thống cần phải được đồng nhất. |
| 3 | Tính khả dụng | Phù hợp nhu cầu | Hệ thống được xây dựng lên cần phải đảm bảo theo sát yêu cầu đề bài: ứng dụng xây dựng lên sẽ hỗ trợ tối đa cho việc quản lý sinh viên và đăng ký tín chỉ cho các sinh viên |
|  |  | Tính dễ sử dụng | Hệ thống cần phải được thiết kế với giao diện đơn giản, giúp người dùng thực hiện ít thao tác nhất khi thực hiện điều hướng đến một nội dung nào đó có trong hệ thống |
|  |  | Khả năng truy cập | Hệ thống cần phải hỗ trợ được người dùng có thể thực hiện đăng nhập tại bất cứ đâu, trên cả 3 nền tảng là Web, Desktop và Android |
| 4 | Tính tin cậy | Trưởng thành | Hệ thống được thiết kế sao cho việc nâng cấp, phát triển các tính năng của hệ thống sẽ được thực hiện một cách dễ dàng nhất |
|  |  | Khả năng chịu lỗi | Khi xảy ra một số vấn đề không mong muốn: số lượng các yêu cầu được gửi lên quá nhiều; có lỗi không mong muốn xảy ra, hệ thống vẫn có thể hoạt động bình thường và không bị gián đoạn |
|  |  | Khả năng phục hồi | Khi hệ thống gặp phải các lỗi nguy hiểm làm hệ thống bị tạm ngưng hoạt động, hệ thống cần phải có khả năng phục hồi và trở lại hoạt động trong thời gian sớm nhất có thể |
| 5 | An toàn thông tin | Bảo mật | Các thông tin có trong hệ thống cần được phân cấp cụ thể những người có thể truy cập tới các thông tin đó:   * Người quản lý của hệ thống có toàn quyền xem các thông tin có trong hệ thống * Giảng viên có thể xem tất cả thông tin và thêm các học phần, lớp học * Sinh viên có thể xem tất cả các thông tin, nhưng chỉ có thể đăng ký học phần và tín chỉ |

# Công nghệ sử dụng

## Danh sách các công nghệ sử dụng

##### Front end

Trong front-end, em không sử dụng các framework hay template có sẵn mà thiết kế hoàn toàn bằng các ngôn ngữ cơ bản để dễ tối ưu hóa và nâng cao hiệu năng.

Danh sách các công nghệ sử dụng tại front-end:

Về ngôn ngữ: HTML, CSS, Javascript

Về thư viện hỗ trợ: Jquery, Jquery UI

##### Back end

Ngôn ngữ lập trình: C#

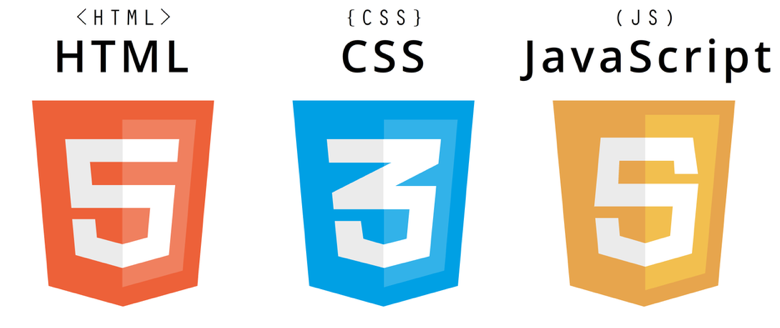
Nền tảng chính: Microsoft .NET Framework

Các công nghệ sử dụng: Microsoft .NET Web API, .NET HTTP, RESTful Webservice, Repository Pattern, Unit of Work, Unity Dependency Injection

##### Database

Sử dụng SQL Server 2017

## HTML, CSS, Javascript



HTML, CSS, Javascript là 3 công nghệ cơ bản của lập trình website

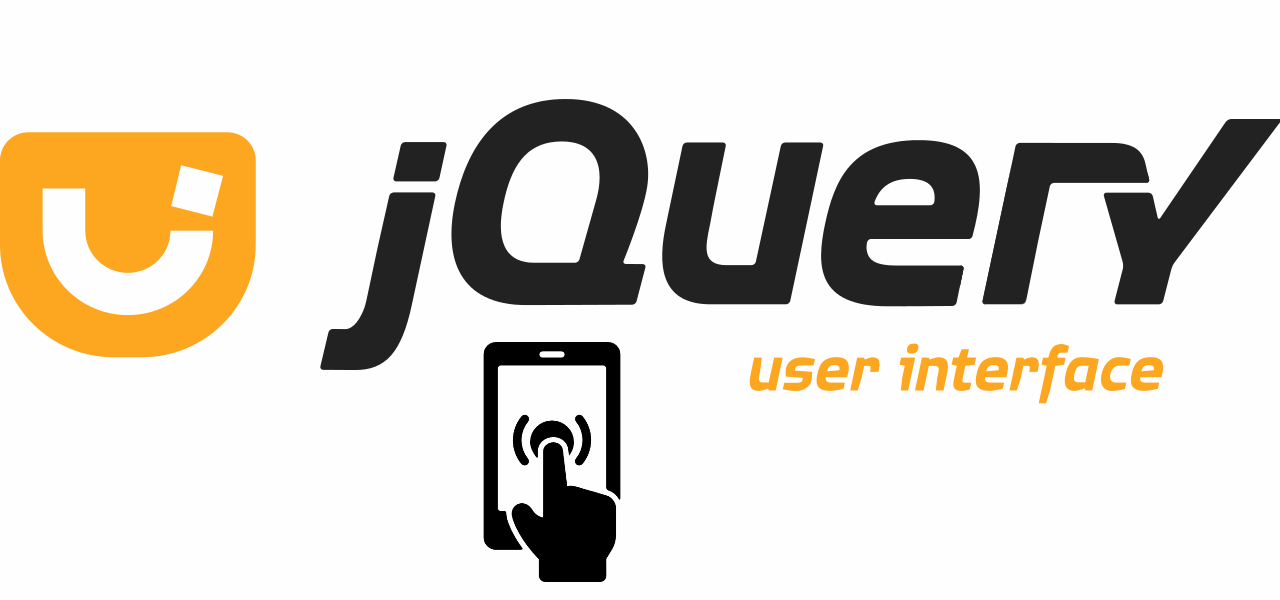
Đây là 3 công nghệ phổ biến và cơ bản nhất để xây dựng website, em sẽ không đi sâu chi tiết vào phần này. Tài liệu tham khảo có thể tìm thấy rõ ràng và chi tiết từ W3C [1].

## JQuery và jQueryUI



##### JQuery

JQuery là một [thư viện JavaScript](https://vi.wikipedia.org/w/index.php?title=Th%C6%B0_vi%E1%BB%87n_JavaScript&action=edit&redlink=1) [đa trình duyệt](https://vi.wikipedia.org/w/index.php?title=%C4%90a_tr%C3%ACnh_duy%E1%BB%87t&action=edit&redlink=1) được thiết kế để đơn giản hóa [lập trình phía máy người dùng](https://vi.wikipedia.org/w/index.php?title=L%E1%BA%ADp_tr%C3%ACnh_ph%C3%ADa_m%C3%A1y_ng%C6%B0%E1%BB%9Di_d%C3%B9ng&action=edit&redlink=1) của [HTML](https://vi.wikipedia.org/wiki/HTML), phát hành vào tháng 1 năm 2006 bởi [John Resig](https://vi.wikipedia.org/w/index.php?title=John_Resig&action=edit&redlink=1). Được sử dụng bởi hơn 52% trong 10.000 truy cập nhiều nhất các trang web, jQuery là [thư viện JavaScript](https://vi.wikipedia.org/w/index.php?title=Th%C6%B0_vi%E1%BB%87n_JavaScript&action=edit&redlink=1) phổ biến nhất được sử dụng ngày nay. [2]



##### JQueryUI

JQueryUI là một bộ các giao diện, hiệu ứng, tiện ích và chủ đề. Được xây dựng trên thư viện JQuery và JavaScript. Sử dụng để xây dựng những trang web có tính tương tác cao [3]. JqueryUI cung cấp thư viện về các tương tác và tiện ích như Draggable, Droppable, Resizeable, Dialog, Menu, Slider…

## C# và .NET Framework

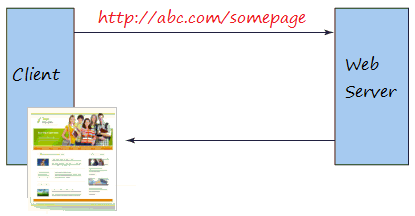


**C#** là một ngôn ngữ lập trình đơn giản, hiện đại, mục đích tổng quát, hướng đối tượng được phát triển bởi Microsoft và được phê chuẩn bởi European Computer Manufacturers Association (ECMA) và International Standards Organization (ISO). Cấu trúc C# khá gần với các ngôn ngữ high-level truyền thống, C và C++, và là một ngôn ngữ lập trình hướng đối tượng [4].

**.NET Framework** là một nền tảng lập trình và cũng là một nền tảng thực thi ứng dụng chủ yếu trên hệ điều hành Microsoft Windows, được phát triển bởi [Microsoft](https://vi.wikipedia.org/wiki/Microsoft). NET framework bao gồm tập các thư viện lập trình lớn, và những thư viện này hỗ trợ việc xây dựng các chương trình phần mềm như lập trình giao diện; truy cập, kết nối [cơ sở dữ liệu](https://vi.wikipedia.org/wiki/C%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u); [ứng dụng web](https://vi.wikipedia.org/wiki/%E1%BB%A8ng_d%E1%BB%A5ng_web); các [giải thuật](https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n), [cấu trúc dữ liệu](https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u); giao tiếp mạng... [5]

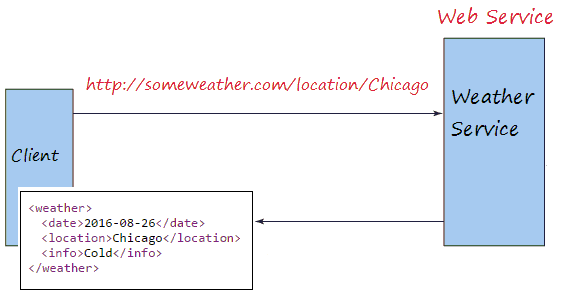
## RESTful Web Service

##### Web service



Hình 17 Web server

Với các web server thông thường, khi chúng ta truy cập server sẽ trả về cho ta nội dung của 1 trang web dưới dạng html, css và javascript.



Hình 18 Web service

Trong khi đó, Web service là 1 dịch vụ Web, khi chúng ta truy cập vào Web service, nó sẽ chỉ trả về cho ta dữ liệu thô, thường dưới dạng JSON hoặc XML.

Ví dụ như trong ảnh trên, khi chúng ta gửi yêu cầu thời tiết Chicago vào Webservice thông tin thời tiết, nó sẽ trả về cho chúng ta dữ liệu là thời tiết của Chicago thay vì cả trang web.

##### RESTful Service [6]

RESTful Web Service là các Web Service được viết dựa trên kiến trúc REST. REST định nghĩa các quy tắc kiến trúc để bạn thiết kế Web services, chú trọng vào tài nguyên hệ thống, bao gồm các trạng thái tài nguyên được định dạng như thế nào và được truyền tải qua HTTP, và được viết bởi nhiều ngôn ngữ khác nhau. REST là một bộ quy tắc để tạo ra một ứng dụng Web Service, mà nó tuân thủ 4 nguyên tắc thiết kế cơ bản sau:

* Sử dụng các phương thức HTTP một cách rõ ràng
* Phi trạng thái
* Hiển thị cấu trúc thư mục như các URls
* Truyền tải JavaScript Object Notation (JSON), XML hoặc cả hai.

1. **Sử dụng các phương thức HTTP một cách rõ ràng**

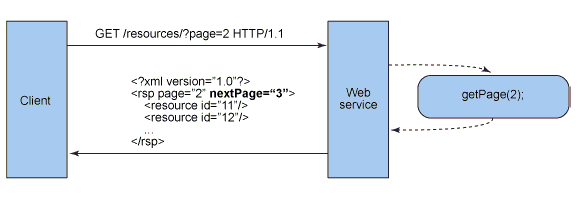
REST đặt ra một quy tắc đòi hỏi lập trình viên xác định rõ ý định của mình thông qua các phương thức của HTTP. Thông thường ý định đó bao gồm lấy dữ liệu, trèn dữ liệu, cập nhập dữ liệu hoặc xóa dữ liệu. Vậy khi bạn muốn thực hiện một trong các ý định trên hãy lưu ý các quy tắc sau:

* Để tạo một tài nguyên trên máy chủ, bạn cần sử dụng phương thức POST.
* Để truy xuất một tài nguyên, sử dụng GET.
* Để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó, sử dụng PUT.
* Để huỷ bỏ hoặc xoá một tài nguyên, sử dụng DELETE.

Chú ý rằng các nguyên tắc ở trên là không bắt buộc, thực tế có thể sử dụng phương thức GET để yêu cầu lấy dữ liệu, chèn, sửa hoặc xóa dữ liệu trên Server. Tuy nhiên REST đưa ra các nguyên tắc ở trên mục đích đưa mọi thứ trở lên rõ ràng và dễ hiểu.

1. **Phi trạng thái**

Một đặc điểm của REST là phi trạng thái (stateless), có nghĩa là nó không lưu giữ thông tin của client. Chẳng hạn bạn vừa gửi yêu cầu để xem trang thứ 2 của một tài liệu, và bây giờ bạn muốn xem trang tiếp theo (sẽ là trang 3). REST không lưu trữ lại thông tin rằng trước đó nó đã phục vụ bạn trang số 2. Điều đó có nghĩa là REST không quản lý phiên làm việc (Session).



Hình 19 Phi trạng thái (Stateless)

Như ở hình trên, Client phải gửi yêu cầu rõ ràng, bao gồm số thự tự của trang cần xem. Như vậy, các thành phần máy chủ phi trạng thái ít phức tạp hơn để thiết kế, viết và phân bổ thông qua máy chủ được cân bằng tải. Dịch vụ phi trạng thái không chỉ hoạt động tốt hơn, nó còn chuyển hầu hết vai trò duy trì trạng thái sang ứng dụng ở máy khách. Trong một dịch vụ mạng RESTful, máy chủ chịu trách nhiệm đưa ra các phản hồi và cung cấp một cách thức cho phép Client duy trì trạng thái ứng dụng của chính nó.

1. **Đưa ra cấu trúc thư mục giống các URI**

REST đưa ra một cấu trúc để người dùng có thể truy cập vào tài nguyên của nó thông qua các URL, tài nguyên ở đây là dữ liệu về sinh viên, môn học, lớp học…

Các địa chỉ REST service cần phải thật trực quan đến mức người dùng dễ đoán. Hãy nghĩ về một địa chỉ (URI) giống như một gợi ý rõ ràng, dễ đoán rằng nó đang trỏ tới cái gì và cung cấp tài nguyên gì. Tóm lại, cấu trúc của một URI nên được đơn giản, có thể dự đoán, và dễ hiểu.

VD link để lấy thông tin sinh viên

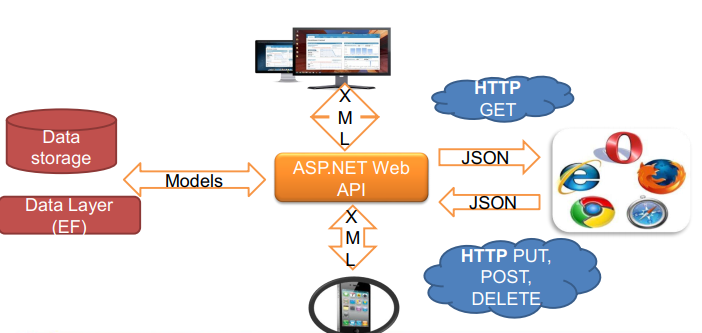
<http://localhost:49336/api/student/id?StudentName=T%E1%BA%A1%20Minh%20Ch%C3%AD>

khi truy cập vào link trên, Service sẽ trả về cho chúng ta thông tin của học sinh có tên Tạ Minh Chí

1. **Truyền tải XML, JSON hoặc cả hai**

Dữ liệu gửi lên hoặc trả về có thể ở dạng XML hoặc JSON hoặc cả 2.

## Microsoft .NET Web API



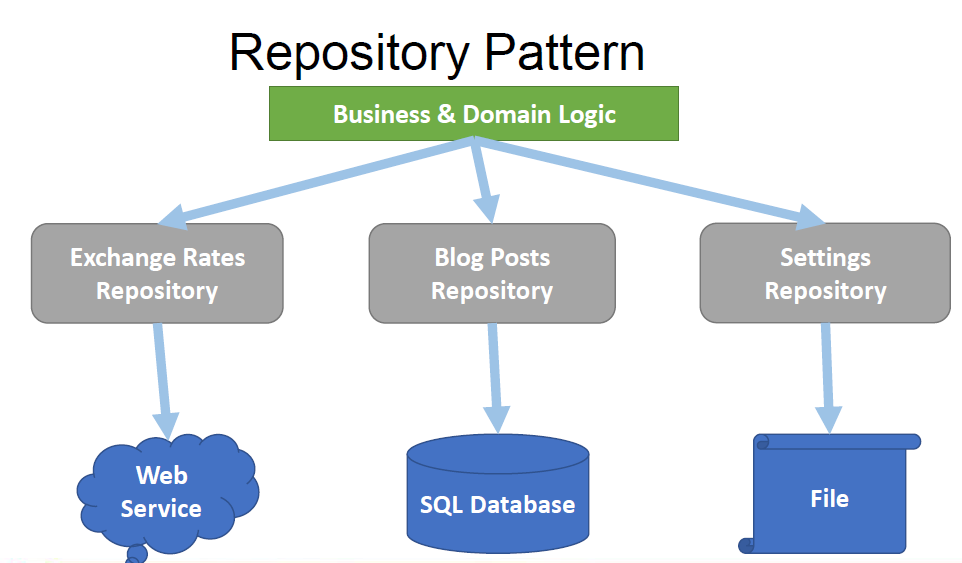
Hình 20 Web API

Đây là một framework mới giúp cho việc xây dựng các HTTP service rất đơn giản và nhanh chóng, Open Source và có thể được sử dụng bởi bất kì client nào hỗ trợ XML, JSON. Hỗ trợ đầy đủ các thành phần HTTP: URI, request/response headers, caching, versioning, content formats, có thể host trong ứng dụng hoặc trên IIS, là kiến trúc lý tưởng cho các thiết bị có băng thông giới hạn như smartphone, tablet và định dạng dữ liệu có thể là JSON, XML hoặc một kiểu dữ liệu bất kỳ [7].

Web API có các ưu điểm:

* Cấu hình hết sức đơn giản khi so với WCF
* Performance cao
* Hỗ trợ RESTful đầy đủ
* Hỗ trợ đầy đủ các thành phần MVC như: routing, controller, action result, filter, model binder, IoC container, dependency injection, unit test
* Open Source

## Repository Pattern



Hình 21 Repository Pattern

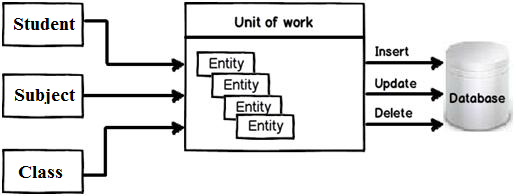
Repository là tầng trung gian giữa Business Service và Data Access. Trong đó Business Service là tầng xử lý nghiệp vụ còn Data Access là tầng truy cập đến cơ sở dữ liệu. Repository hầu hết được sử dụng những chỗ cần điều chỉnh dữ liệu trước khi truyền xuống tầng data hoặc truyền lên trên business logic [8].

Lợi ích của việc sử dụng Repository Pattern:

* Tập trung hóa được các logic về xử lý dữ liệu hoặc business logic.
* Dễ dàng thực hiện các unit test
* Khi thay đổi logic của tầng data hoặc business logic, không cần thay đổi Repository

Khi chúng ta làm việc trong một khối kiến trúc, các project của chúng ta phải độc lập giữa tất cả các lớp. Hay nói cách khác lớp Controller và DataAccess phải độc lập nhau. Nếu chúng ta để chúng kết hợp chặt chẽ thì mọi thay đổi ở lớp DataAccess sẽ làm thay đổi mã Controller. Do vậy, **Reposotory** được sinh ra giữa tầng **Data Access** và tầng **Bussiness** (nghiệp vụ) của ứng dụng. Mô hình này giúp bảo vệ ứng dụng khỏi những thay đổi từ kho dữ liệu và tạo điều kiện thuận lợi cho việc kiểm nghiệm.

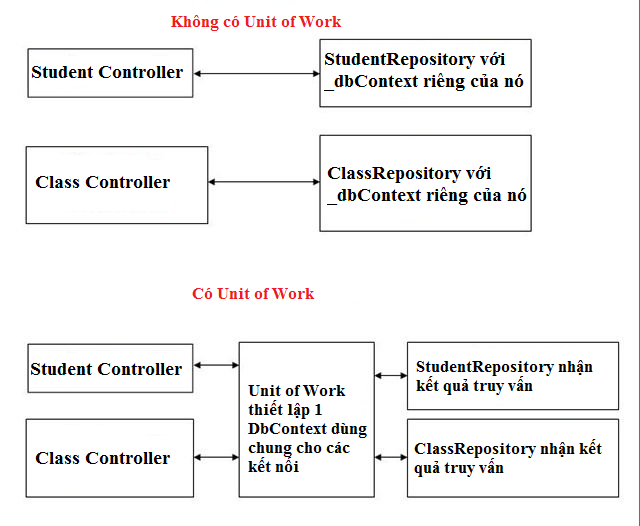
## Unit of work



Hình 22 Unit of work

Unit of Work được sử dụng với Repository Pattern để đảm bảo toàn vẹn dữ liệu cũng như tránh có lỗi xảy ra.

Nó được sử dụng để đảm bảo nhiều hành động như insert, update, delete...được thực thi trong cùng một transaction thống nhất. Nói đơn giản hơn, nghĩa là khi một hành động của người dùng tác động vào hệ thống, tất cả các hành động như insert, update, delete...phải thực hiện xong thì mới gọi là một transaction thành công. Gói tất cả các hành động đơn lẻ vào một transaction để đảm bảo tính toàn vẹn dữ liệu.



Hình 23 Chức năng của Unit of Work

Khi có Unit of Work, các bộ truy vấn sẽ được bảo đảm thực hiện toàn vẹn, không xảy ra tình trạng truy vấn chỉ thực hiện được một phần rồi bị lỗi, gây ra lỗi trong cơ sở dữ liệu

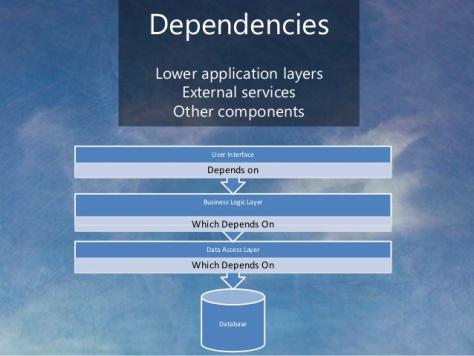
## Unity Dependency Injection

##### SOLID

SOLID là viết tắt của 5 chữ cái đầu trong 5 nguyên tắc thiết kế hướng đối tượng, giúp cho developer viết ra những đoạn code dễ đọc, dễ hiểu, dễ triển khai. 5 nguyên tắc đó bao gồm:

* **S**ingle responsibility priciple (SRP) : Một class chỉ nên giữ 1 trách nhiệm duy nhất (Chỉ có thể sửa đổi class với 1 lý do duy nhất)
* **O**pen/Closed principle (OCP): Có thể thoải mái mở rộng 1 class, nhưng không được sửa đổi bên trong class đó (open for extension but closed for modification).
* **L**iskov substitution principe (LSP): Trong một chương trình, các object của class con có thể thay thế class cha mà không làm thay đổi tính đúng đắn của chương trình
* **I**nterface segregation principle (ISP): Thay vì dùng 1 interface lớn, ta nên tách thành nhiều interface nhỏ, với nhiều mục đích cụ thể
* **D**ependency inversion principle (DIP): Các module cấp cao không nên phụ thuộc vào các modules cấp thấp. Cả 2 nên phụ thuộc vào abstraction. Interface (abstraction) không nên phụ thuộc vào chi tiết, mà ngược lại. (Các class giao tiếp với nhau thông qua interface, không phải thông qua implementation.)

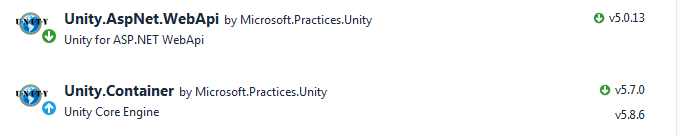
##### Dependency Injection



Hình 24 Mô hình Dependency Injection

Dependency Injection là một kỹ thuật sử dụng để phát triển ứng dụng một cách độc lập. Độc lập nghĩa là một mô-đun của ứng dụng phải là duy nhất và không bị phụ thuộc và các mô-đun khác.

##### Unity



Hình 25 Unity ASPNet Web API và Unity Container

Unity ASPNet Web API và Unity Container là 2 packages mở rộng của Visual Studio, cung cấp các công cụ để triển khai Dependency Injection trong hệ thống. Cách triển khai như thế nào sẽ được trình bày cụ thể trong chương 5.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

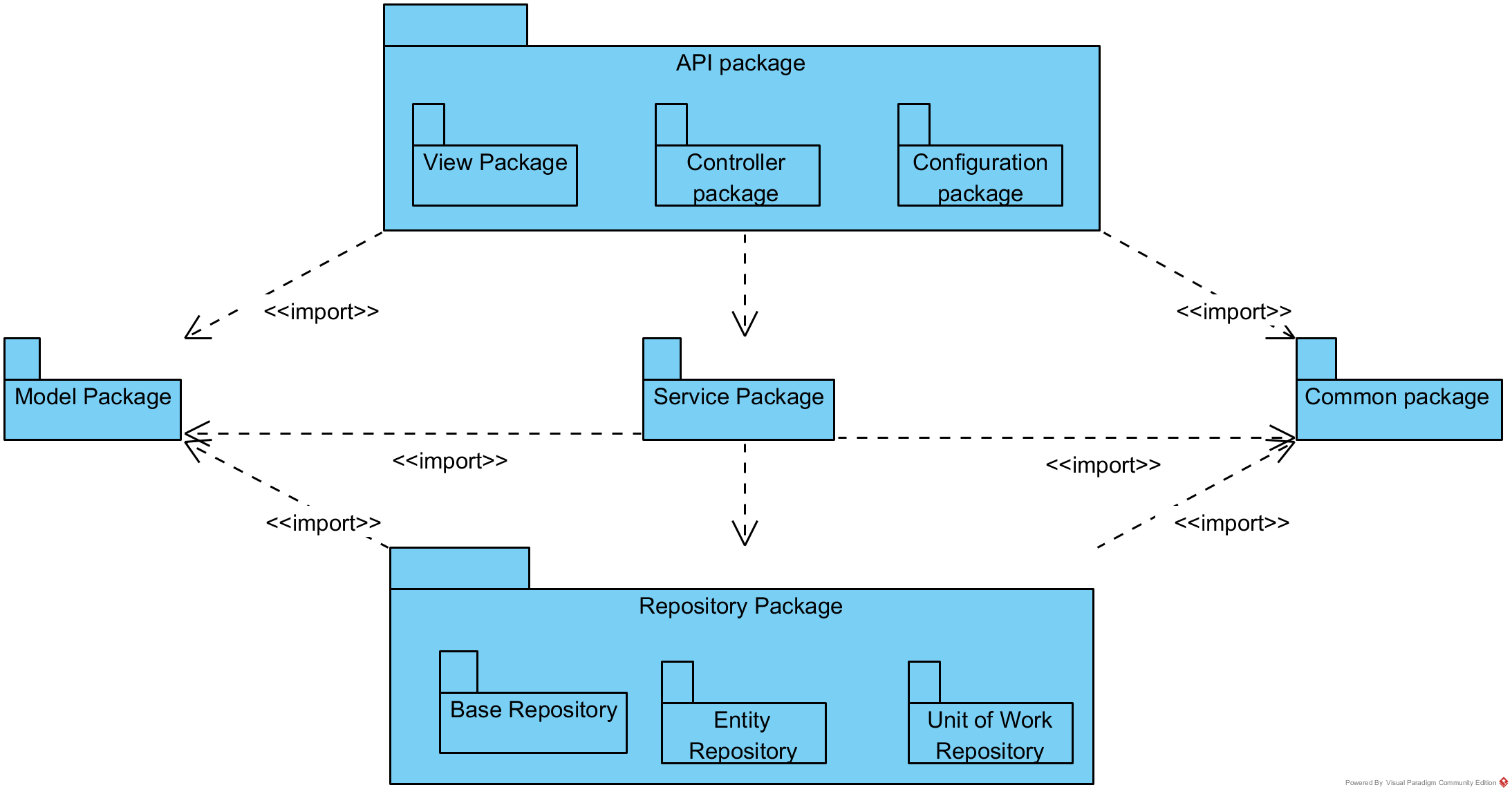
### Lựa chọn kiến trúc

Phần mềm sử dụng kiến trúc 5 tầng dựa trên .NET Web API và Repository Pattern bao gồm:

Bảng 17 Mô tả Kiến trúc chương trình

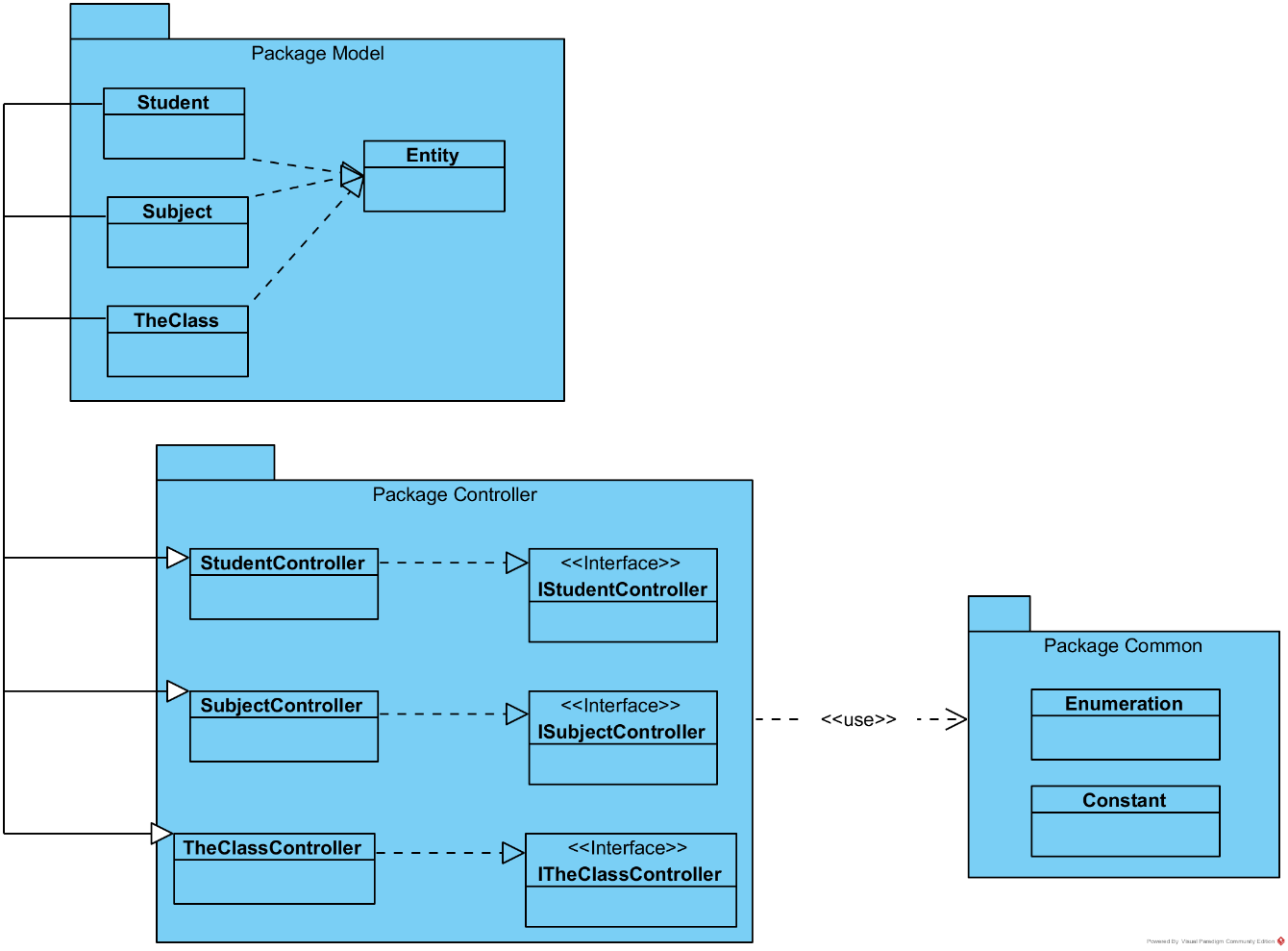
|  |  |  |
| --- | --- | --- |
| Tầng | Chức năng | Các thành phần |
| View & Control | Hiển thị giao diện, nhận yêu cầu người dùng và điều hướng yêu cầu người dùng | Folder View: chứa các file hiển thị, css, javascript |
| Class Controller: Điều hướng cho các yêu cầu từ phía client gửi lên, thường là gọi đến tầng Service |
| WebConfig: Chứa cấu hình về các trang web. VD: map các trang và url, địa chỉ kết nối đến CSDL, cấu hình trang chủ,... |
| Common | Chứa các hằng số, các kiểu định nghĩa | Class Constant: chứa các hằng số |
| Class Enumeration: chứa các kiểu Enum như Giới tính, Khoa viện,... |
| Model | Chứa các lớp khuôn mẫu để đổ dữ liệu | Class Student: khuôn mẫu chứa các dữ liệu của sinh viên |
| Class Subject: khuôn mẫu chứa các dữ liệu về môn học |
| Class TheClass: Khuôn mãu chứa các dữ liệu về lớp học |
| Class User: Khuôn mẫu chứa các dữ liệu về người dùng |
| Service | Chứa các lớp thực thi nghiệp vụ | Class StudentService: chứa các phương thức xử lý nghiệp vụ cho đối tượng học sinh như Thêm, sửa ,xóa... |
| Class SubjectService và TheClassService: tương tự |
| Ngoài ra mỗi class còn có 1 interface để tăng tính tương tác và đảm bảo Dependency inversion |
| Repository | Liên kết với cơ sở dữ liệu, thực thi các truy vấn để | Folder DatabaseAccess: Chứa các class thực thi kết nối tới database |
| Folder Base: chứa các class thực thi truy vấn chung |
| Class StudentRepository: chứa các phương thức truy vấn đến CSDL của đối tượng học sinh như Thêm, sửa, xóa |
| Class SubjectRepository và TheClassRepository: tương tự như StudentRepository |
| Ngoài ra mỗi class còn có 1 interface để tăng tính tương tác và đảm bảo Dependency inversion |

### Thiết kế tổng quan

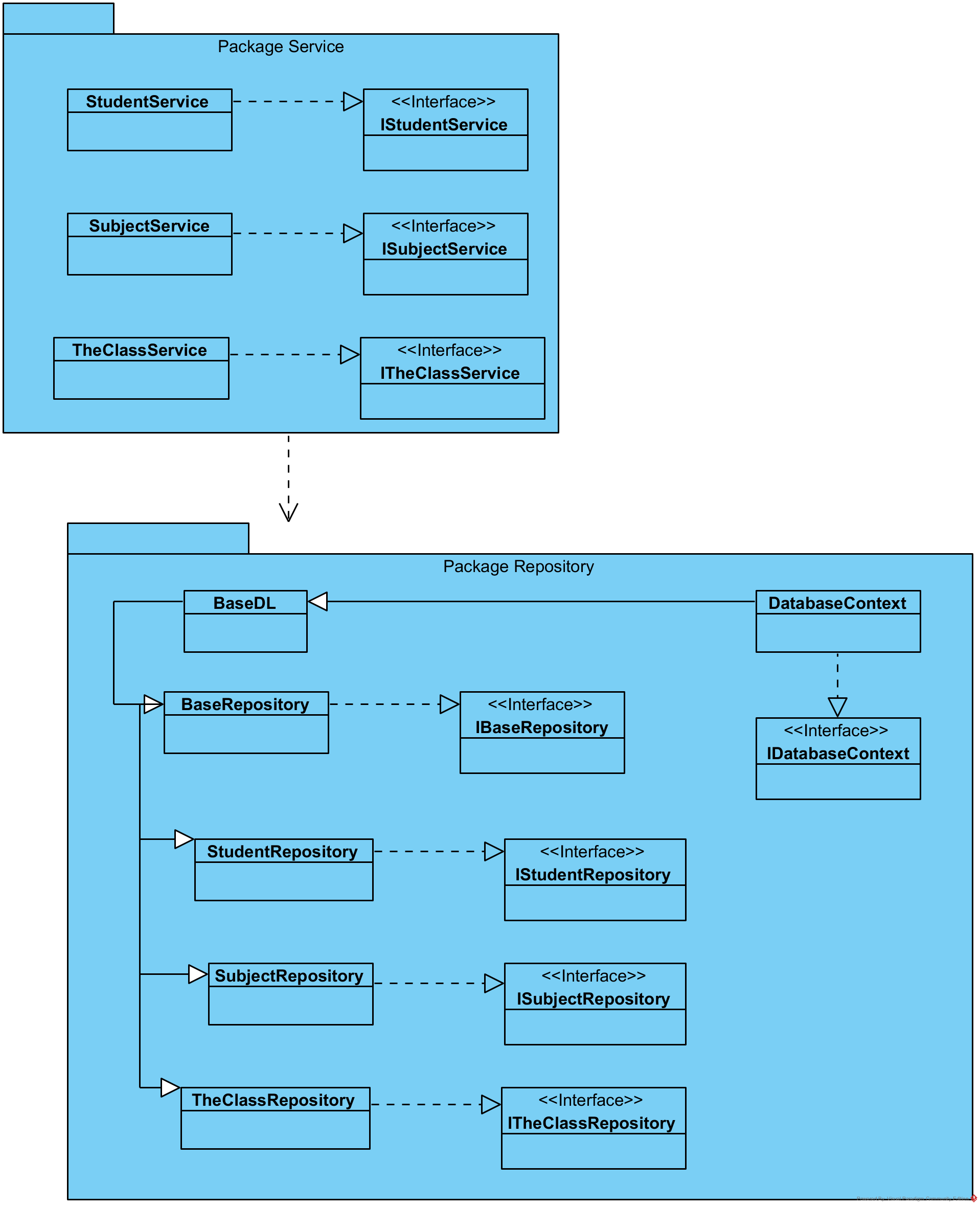


Hình 26 Package diagram

### Thiết kế chi tiết gói



Hình 27 Biểu đồ chi tiết package Controller, Model và Common



Hình 28 Biểu đồ chi tiết package Service và Repository

BaseDL: Chứa các phương thức thực thi database

BaseRepository: Chứa các phương thức thêm, sửa, xóa… cho thực thể

StudentRepository: Chứa các phương thức thêm, sửa, xóa… cho đối tượng sinh viên

DatabaseContext: chứa các phương thức kết nối, ngắt kết nối với CSDL

## Thiết kế chi tiết

### Thiết kế giao diện

Giao diện được thiết kế trên nền tảng web. Vì là một hệ thống nghiệp vụ nên giao diện cần đơn giản, gọn gàng và tiện ích. Ngoài ra cũng cần phải gọn nhẹ để đảm bảo lưu lượng truy cập.

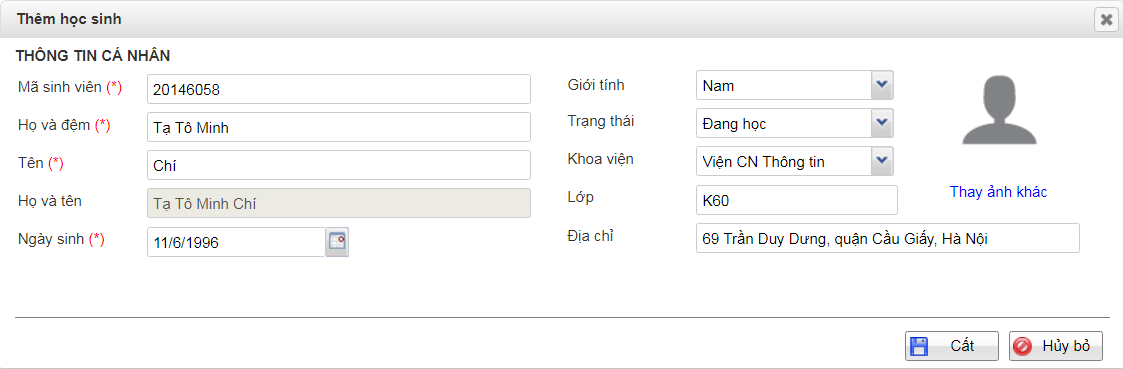
Nền tảng chính là website trên máy tính để bàn. Kích thước màn hình tối thiểu 1024x768. Giao diện cũng được thiết kế reponsive để tương thích với nhiều kích cỡ màn hình khác nhau.

Các thành phần như bảng biểu, ô nhập text, button,.. được thiết kế theo nguyên mẫu của hệ thống quản lý sinh viên hiện tại, ngoài ra có tham khảo thêm tại hệ thống Quản lý trường học của Cty MISA. Dưới đây là một số thành phần cơ bản của giao diện.



Hình 29 Form đăng nhập

Form đăng nhập được thiết kế gọn gàng, đơn giản.



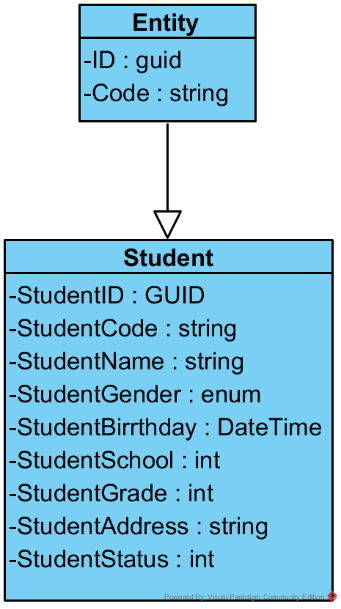
Hình 30 Form thêm thông tin sinh viên

Form Thêm sinh viên được thiết kế ngắn gọn, đủ ý, sẽ hiện ra khi người dùng click vào chức năng thêm sinh viên. Form có đầy đủ các thông tin của sinh viên cần thêm.

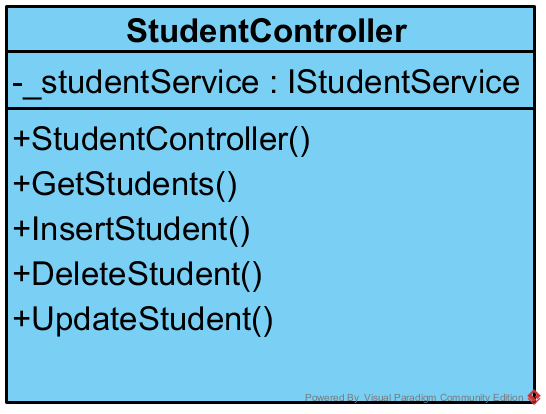
Đối với Form sửa sinh viên cũng tương tự.

### Thiết kế lớp

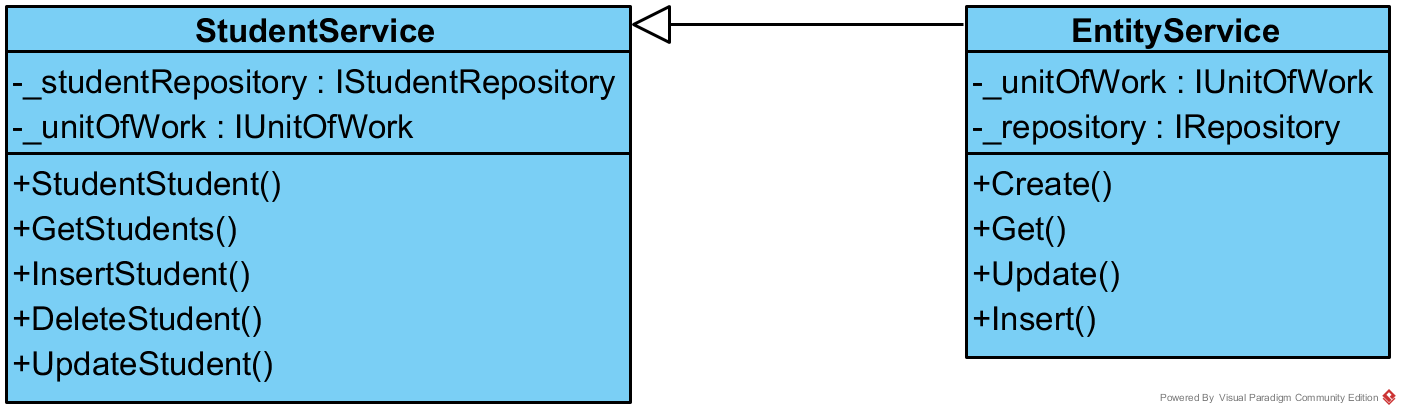
Biểu đồ một số lớp quan trọng



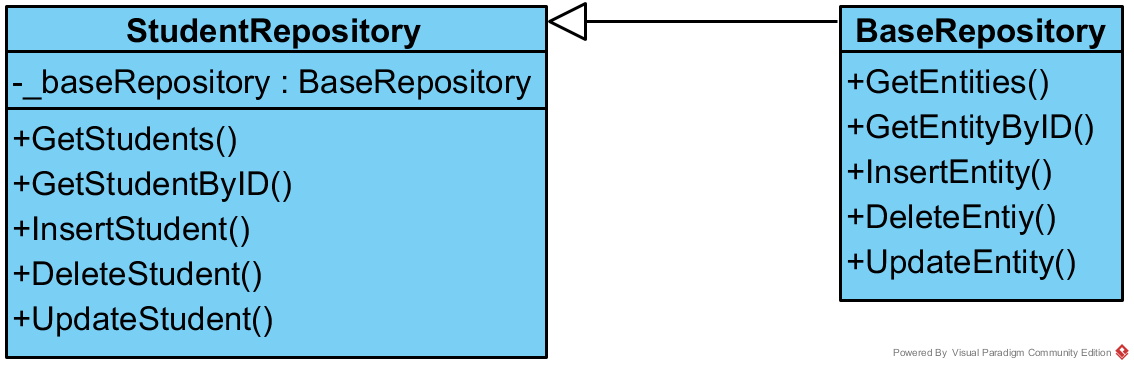
Hình 31 Biểu đồ lớp của lớp Student



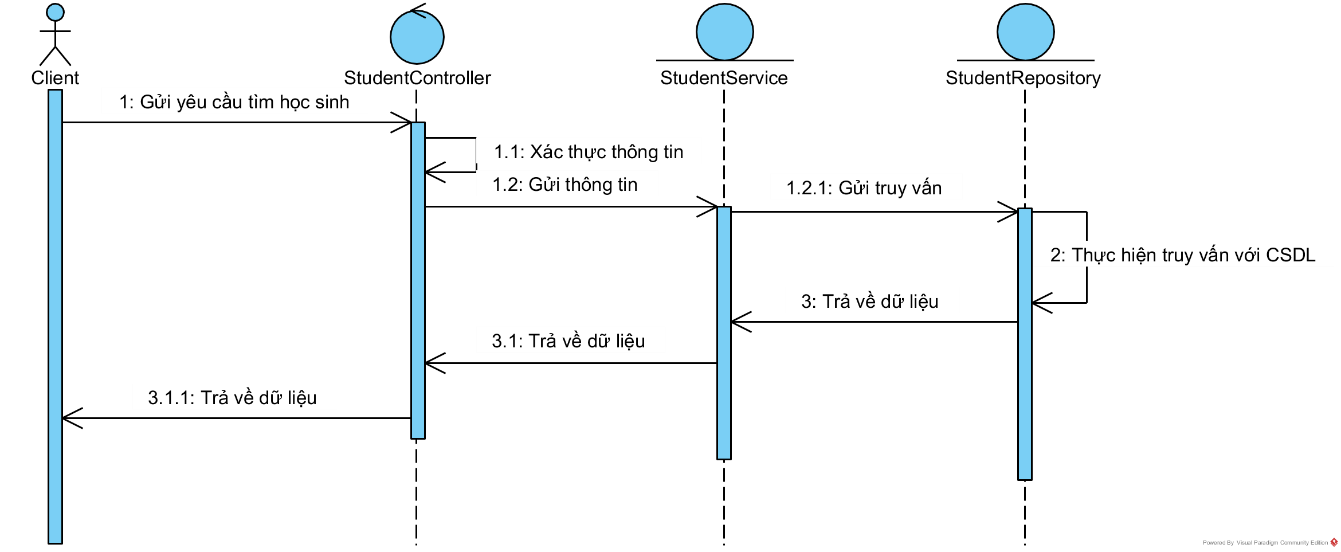
Hình 32 Biểu đồ lớp Student Controller



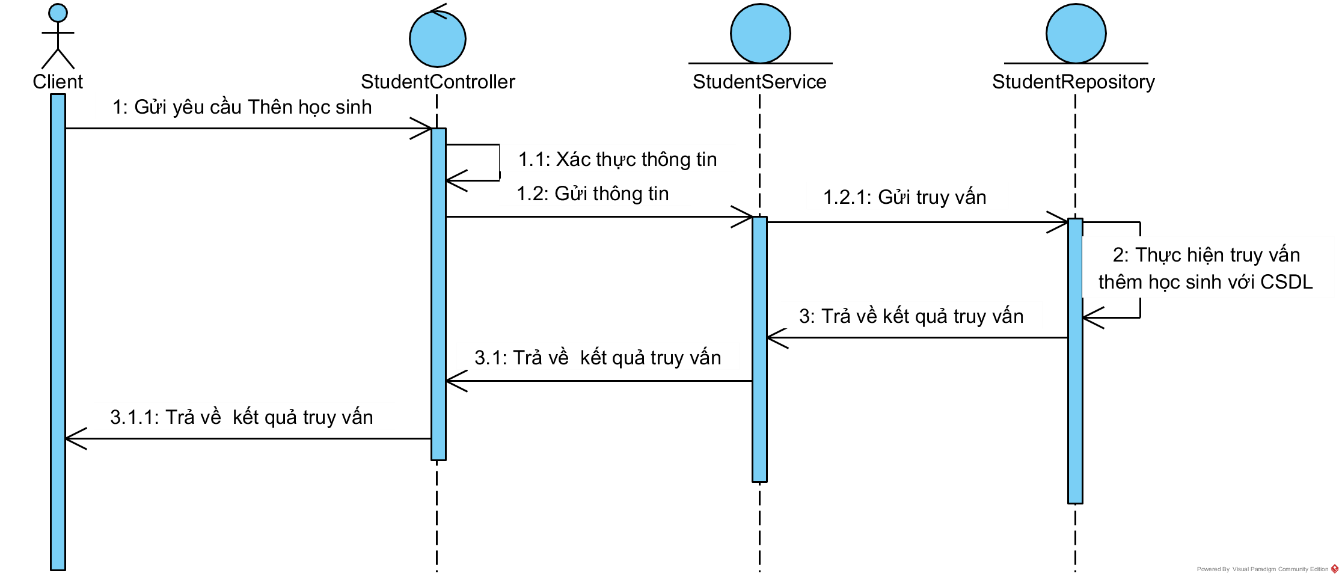
Hình 33 Biểu đồ lớp StudentService



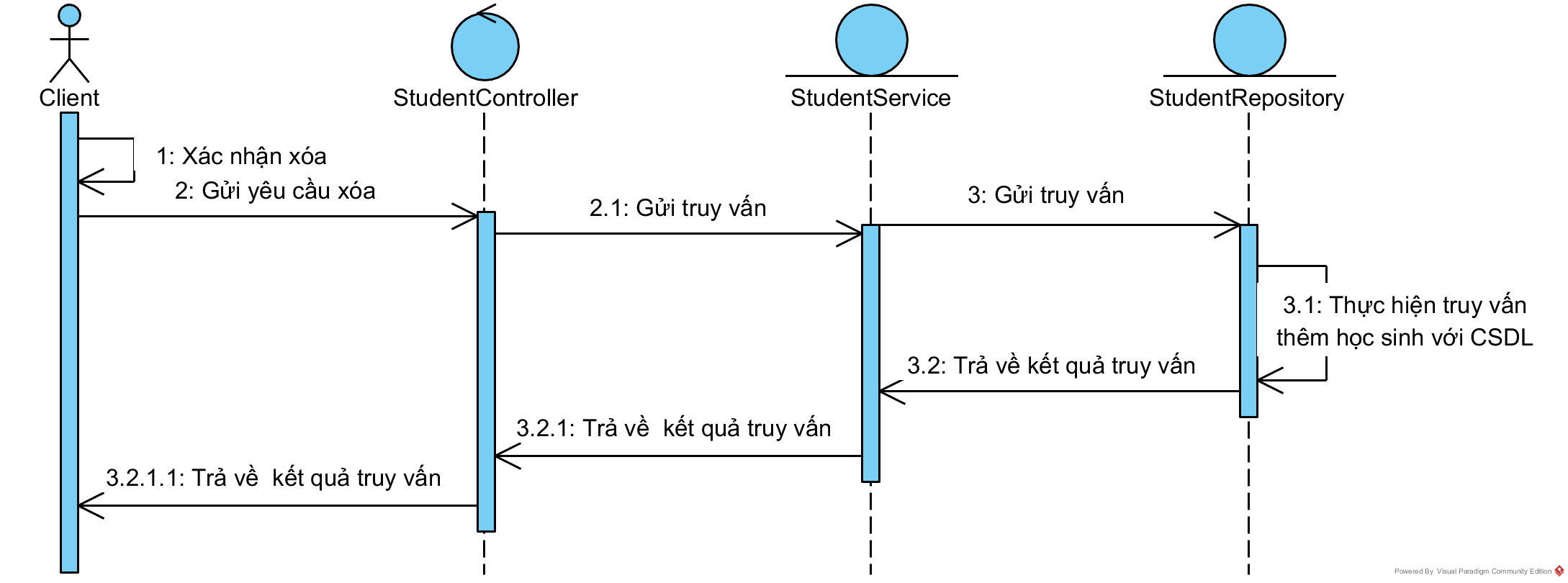
Hình 34 Biểu đồ lớp Student Repository



Hình 35 Biểu đồ tuần tự tìm sinh viên

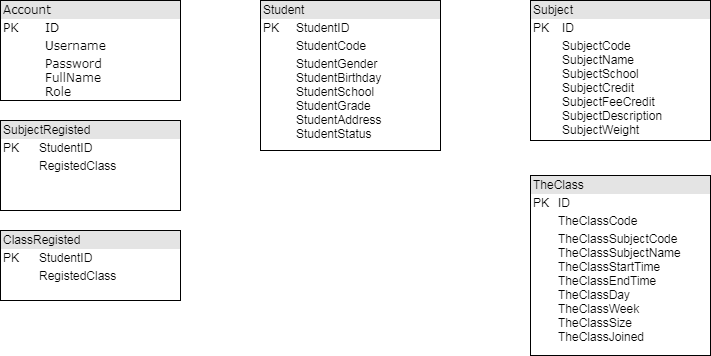


Hình 36 Biểu đồ tuần tự thêm học sinh



Hình 37 Biểu đồ tuần tự Xóa học sinh

### Thiết kế cơ sở dữ liệu



Hình 38 Biểu đồ Cơ sở dữ liệu

Các bảng không có liên kết với nhau, lý do sẽ được giải thích ở chương 5.

Dưới đây là mô tả cho một số bảng quan trọng

Bảng 18 Mô tả CSDL bảng Student

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên cột | Kiểu dữ liệu | Mô tả |
| 1 | StudentID | uniqueidentifier | ID được lưu dưới dạng Uniqueidentifier để tăng tốc độ truy vấn |
| 2 | StudentCode | varchar(8) | Là 1 chuỗi 8 ký tự, VD 20141234 |
| 3 | StudentName | nvarchar(50) | Là 1 chuỗi ký tự, VD Lê Trọng An |
| 4 | StudentGender | int | 0 ứng với Nữ, 1 ứng với Nam |
| 5 | StudentSchool | int | mỗi số ứng với 1 khoa viện |
| 6 | StudentGrade | int | tương ứng số khóa sinh viên đang học |
| 7 | StudentAddress | nvarchar(200) | Địa chỉ của sinh viên |
| 8 | StudentStatus | int | 1 ứng với Đang học, 2: Đã thôi học, 3: Bảo lưu |

Bảng 19 Mô tả CSDL bảng TheClass

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên cột | Kiểu dữ liệu | Mô tả |
| 1 | TheClassID | uniqueidentifier | ID của lớp học |
| 2 | TheClassCode | varchar(10) | Mã lớp |
| 3 | TheClassSubjectCode | varchar(10) | Mã học phần |
| 4 | TheClassSubjectName | nvarchar(100) | Tên học phần |
| 5 | TheClassStartTime | int | Thời gian bắt đầu |
| 6 | TheClassEndTime | int | Thời gian kết thúc |
| 7 | TheClassDay | varchar(100) | Ngày học |
| 8 | TheClassWeek | varchar(100) | Tuần học |
| 9 | TheClassSize | int | Sĩ số tối đa |
| 10 | TheClassJoined | int | Đã đăng ký |

Bảng 20 Mô tả CSDL bảng Account

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên cột | Kiểu dữ liệu | Mô tả |
| 1 | ID | uniqueidentifier | ID của tài khoản |
| 2 | Username | varchar(32) | Tên tài khoản |
| 3 | Password | varchar(32) | Mật khẩu |
| 4 | Fulllname | nvarchar(50) | Tên người dùng |
| 5 | Role | int | Vai trò của người dùng |

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Bảng 21 Thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Visual Studio 2015 | https://www.visualstudio.com/ |
| Nền tảng | .NET Framework 4.5 | https://www.microsoft.com/en-us/download/details.aspx?id=30653 |
| Quản trị CSDL | SQL Server 2017 | https://www.microsoft.com/en-us/sql-server/sql-server-2017 |
| Server | IIS 10.0 Express | https://www.microsoft.com/en-us/download/details.aspx?id=48264 |
| Hiển thị website | Google Chrome | https://www.google.com/intl/vi\_vn/chrome/ |
| Gửi request | Postman 6.0.10 | https://www.getpostman.com/ |
| Vẻ biểu đồ | Visual Paradigm CE 15.0 | https://www.visual-paradigm.com/download/ |

### Kết quả đạt được

Kết quả đạt được là 1 hệ thống website gôm 4 module chính: Đăng nhập, quản lý Thông tin, Quản lý Lớp học, Quản lý Học phần

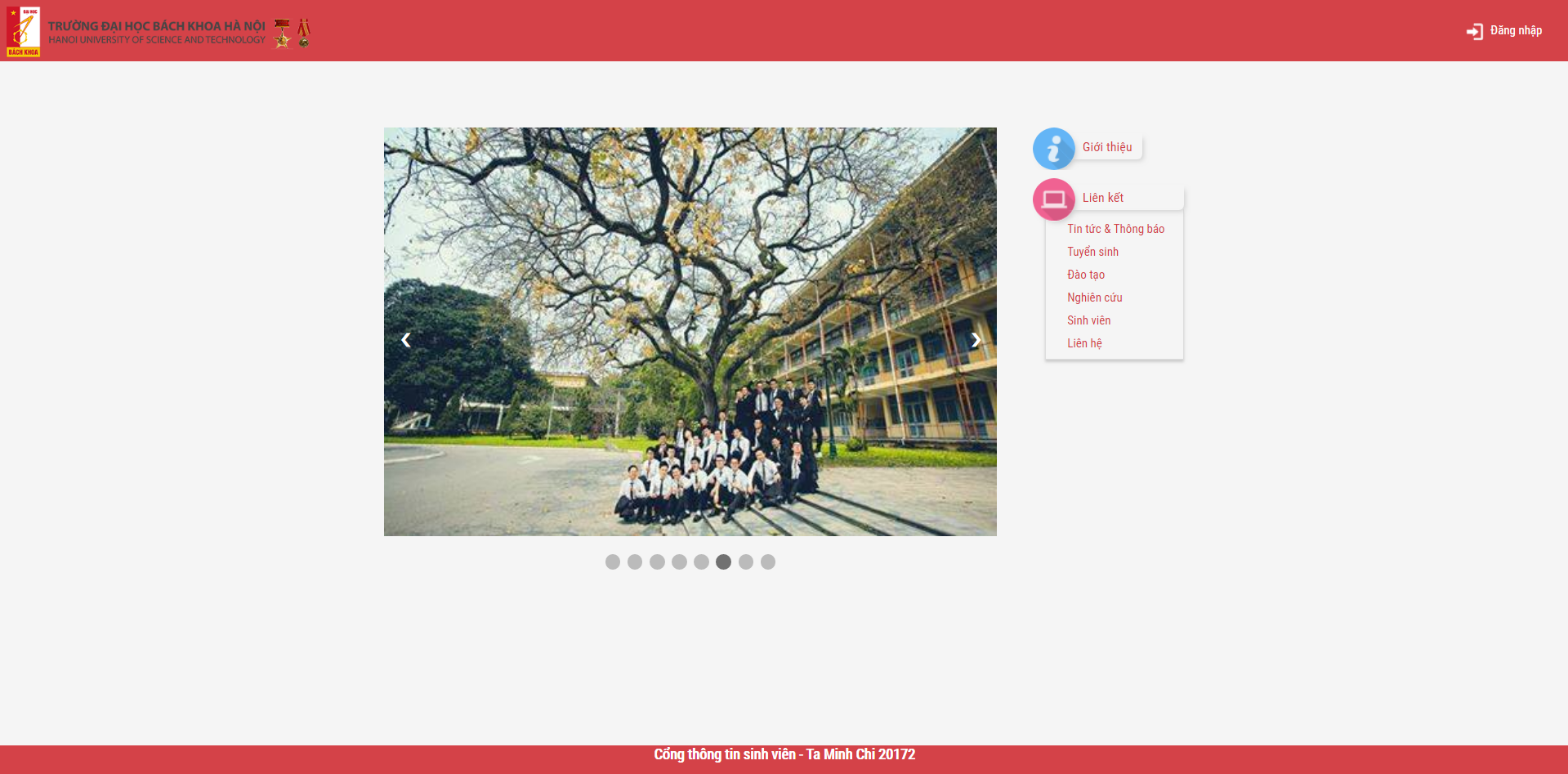
Bảng thống kê chi tiết

Bảng 22 Thống kê kết quả đạt được

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Thông tin | Số lượng | Ghi chú |
| 1 | Package | 5 |  |
| 2 | Số lớp | 23 |  |
| 3 | Số Interface | 15 |  |
| 4 | Dung lượng mã nguồn | 39MB | Dung lượng project trên Visual Studio |
| 5 | Số module | 3 |  |
| 6 | Số webpage | 5 |  |
| 7 | Kích cỡ dữ liệu | 18MB |  |

### Minh hoạ các chức năng chính

Sau khi truy cập vào trang web, người dùng sẽ đơcj đưa đến giao diện trang chủ

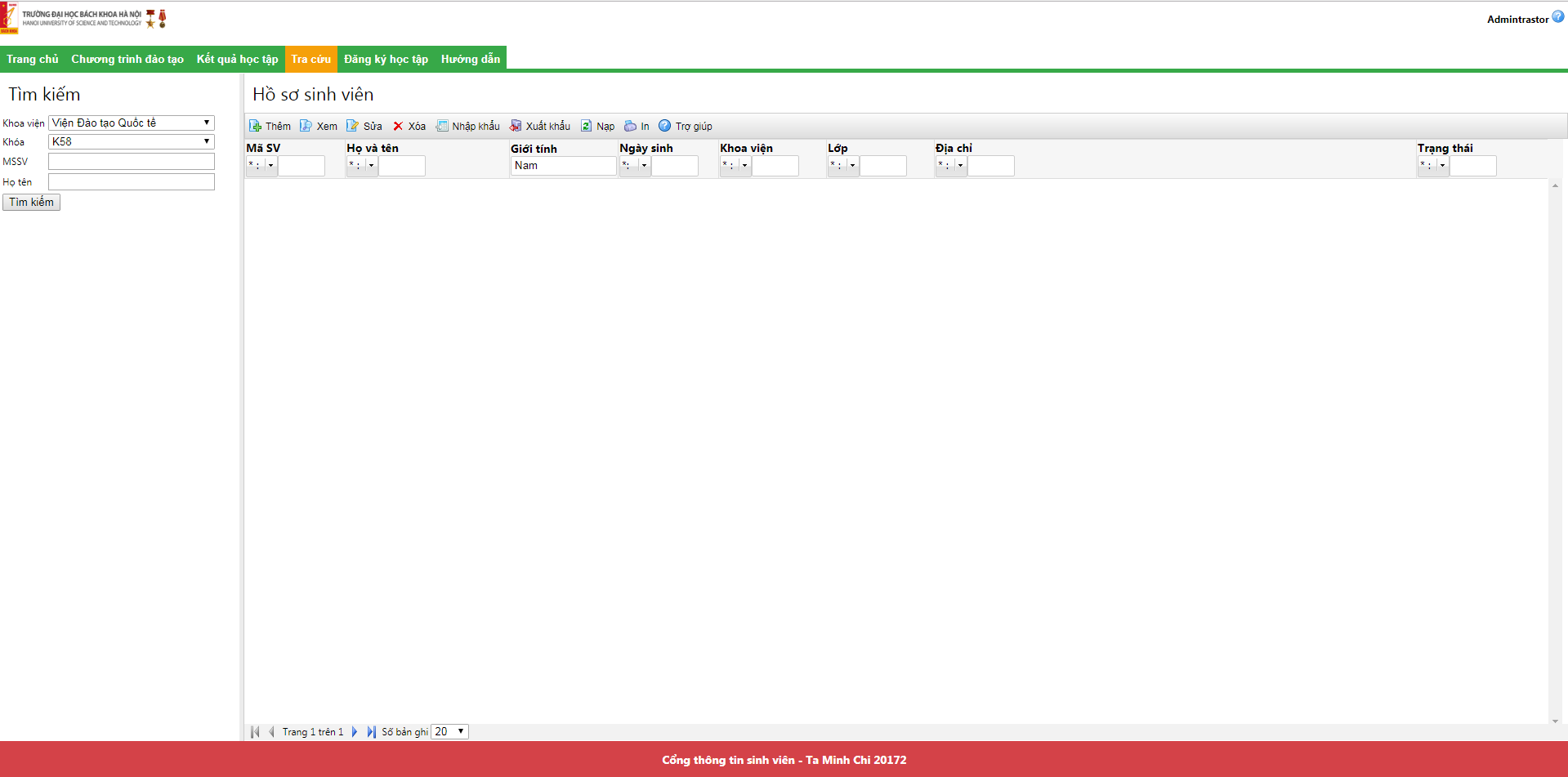


Sau đó người dùng click vào nút **Đăng nhập** ở bên trên, sẽ xuất hiện form đăng nhập

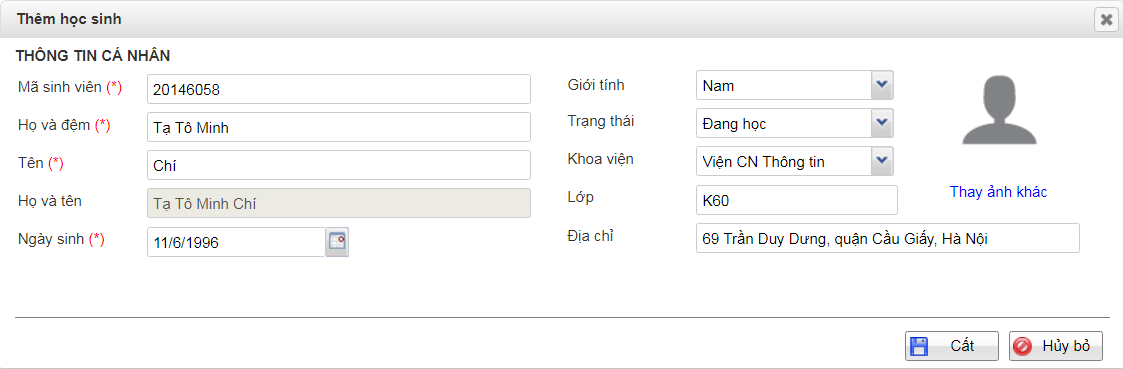


Sau khi đăng nhập, người dùng có thể sử dụng các chức năng quản lý như Quản lý sinh viên. Quản lý lớp học, Quản lý Học phần

Giao diện trang quản lý sinh viên ( các trang quản lý lớp học, quản lý học phần tương tự)



Bảng “Thêm Thông tin” sinh viên ( Bảng “Sửa Thông tin” tương tự)



## Triển khai

Hệ thống được triển khai trên server IIS ảo trên localhost.

Các truy cập được tiến hành từ Browser hoặc Postman

# Các giải pháp và đóng góp nổi bật

Trong chương này, em xin được tóm lược lại một số đóng góp cơ bản của đồ án và những phần mà em cảm thấy quan trọng nhất khi thực hiện đồ án. Về cơ bản, trong đồ án này, em đã thay thế mô hình MVC truyền thống bằng cách sử dụng WEB API để cải tiến về mặt hiệu năng đối với các hệ thống có yêu cầu về số lượng người dùng truy cập lớn. Sau đây là một số tóm lược về các đóng góp cơ bản.

## .NET WEB API được ứng dụng như thế nào trong Project

Không giống như các website PHP hay các website sử dụng framework như WordPress, Magneto... rất phổ biến và có thể tìm thấy mã nguồn ngập tràn trên mạng, .NET Web API chỉ mới được tung ra vài năm gần đây và cũng không phổ biến vì tính phức tạp, chỉ phù hợp cho các ứng dụng nghiệp vụ và các hệ thống lớn của nó. Đây là một nền tảng khá mới, còn rất ít tài liệu tham khảo, chủ yếu là tài liệu của Microsoft và các trang nước ngoài khiến cho việc tiếp cận tài liệu của em rất khó khăn.

Hầu hết các tài liệu đều chỉ mang tính chất giới thiệu hoặc minh họa, việc áp dụng nó vào một dự án thực tế như Quản lý sinh viên là rất khó. Nhưng rất may mắn là trong quãng thời gian thực tập tại MISA, em có cơ hội tiếp cận với những công nghệ mới như này, và được sự tận tình hướng dẫn của các anh chị nhiều kinh nghiệm ở đây. Em đã tháo gỡ những khó khăn trong chuyên môn để hoàn thành đồ án này.

Để nói đến công dụng lớn nhất của Web API, chúng ta có thể hiểu đơn giản là thay vì cung cấp cho người dùng toàn bộ nội dung trang web từ giao diện, hình ảnh, dữ liệu, xử lý dữ liệu như Web Server truyền thống, API Web Service chỉ thực hiện công việc xử lý dữ liệu. Nó nhận đầu vào là dữ liệu (XML hoặc JSON hoặc cả 2), sau đó trả về dữ liệu cho người dùng, không hề gửi về một dòng HTML, CSS, JS nào cả. Chính vì thế chúng ta hoàn toàn có thể chia nhỏ 1 hệ thống thành nhiều API khác nhau VD trong 1 trang Quản lý sinh viên sẽ có các API Quản lý thông tin, Quản lý học phần, Quản lý lớp,... hoặc chúng ta có thể chia nhỏ nó ra thành nhiều API nữa như Thêm thông tin, sửa thông tin, xóa thông tin,... và các API này hoàn toàn có thể được tách ra hoặc gộp vào với nhau, triển khai trên cũng 1 service hoặc nhiều service, chúng ta hoàn toàn có thể từ 1 view gọi đến các Web Service khác nhau như khi vào tab Quản lý thông tin, ta gọi đến Service quản lý thông tin và lấy thông tin sinh viên về bằng 1 truy vấn AJAX với đường link <http://sis.hust.edu.vn/student/GetStudent>, còn khi vào tab Quản lý học phần, ta lại lấy thông tin học phần bằng cách gọi 1 truy vấn AJAX khác vào đường link <http://sis.hust.edu.vn/subject/GetSubject>.

Việc này đảm bảo 2 yếu tố: thứ nhất là khi service đăng ký học phần bị sập, service sinh viên vẫn hoạt động bình thường, thứ 2 là hoàn toàn có thể gọi truy vấn AJAX từ ứng dụng di động, desktop hay thậm chí là các hệ thống nhúng nên nếu muốn tạo ra một hệ thống đa nền tảng, chúng ta chỉ cần viết lại phần giao diện cho mỗi loại thiết bị còn phần hệ thống hoàn toàn không ảnh hưởng. Như vậy trong tương lai, khi muốn phát triển thêm ứng dụng cho các nền tảng mobile như iOS, Android, chúng ta có thể tái sử dụng

## Tại sao phải áp dụng kiến trúc REST?

Khi chúng ta gọi AJAX, chúng ta chỉ truyền lên dữ liệu và địa chỉ API, vậy làm sao hệ thống biết gọi đến hàm nào ở trong Controller? Đó chính là lúc kiến trúc REST phát huy tác dụng, các hàm sẽ được gọi theo phương thức và tham số đầu vào của nó.   
Ví dụ khi gửi lên request GET và không có tham số đầu vào, hệ thống sẽ gọi đến phương thức GetStudents() được cấu hình với phương thức GET và không có tham số đầu vào, còn nếu em vẫn gửi lên request GET với tham số là studentID, hệ thống sẽ gọi đến phương thức GetStudentByStudentID(GUID studentID) đã được cấu hình với phương thức GET và tham số studentID.

Tất nhiên, 2 phương thức với cùng 1 kiểu request và 1 kiểu tham số sẽ sinh ra lỗi.

Như ở đoạn code bên dưới, cả 2 đều là giao thức Get nhưng khác nhau ở tham số đầu vào, 1 phương thức không có tham số còn 1 phương thức có tham số.





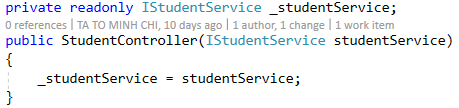
## Unit of Work có đóng góp nổi bật gì?

Chúng ta đã quen với khái niệm Transection (phiên làm việc) trong SQL, thì Unit of Work (đơn vị công việc) cũng tương tự như thế, nhưng mà tại tầng Reponsitory.   
Unit of Work sẽ đảm bảo cho các truy vấn diễn ra theo đúng quy trình, nếu có sai sót sẽ hủy ngay cả quá trình thực thi đấy.

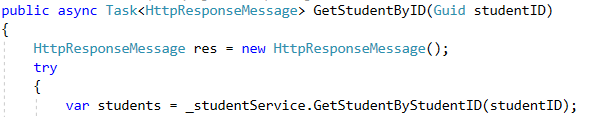
Ngoài ra với các công nghệ cũ, mỗi request sẽ tạo ra 1 kết nối DB của riêng mình, khiến cho khi đang truy vấn thì request khác sẽ không thể kết nối được. Còn Unit of Work sẽ tập hợp các request thành một đơn vị công việc chung, thực hiện trên cùng 1 kết nối.

## Unity Dependency Injection giúp ích gì cho chúng ta?

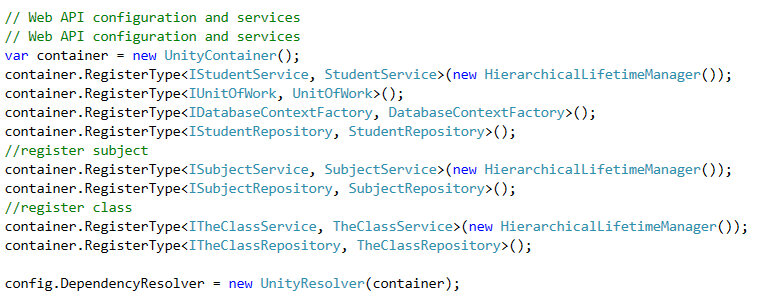
Trong đoạn code, em sử dụng 1 biến kiểu Interface để tham chiếu đến tầng Service



Nhưng em lại hoàn toàn có thể dùng nó để tham chiếu đến các phương thức của lớp StudentService



Đó là nhờ vào Unity Container, giúp cấu hình liên kết giữa Interface và Class, cấu hình như đoạn code bên dưới.

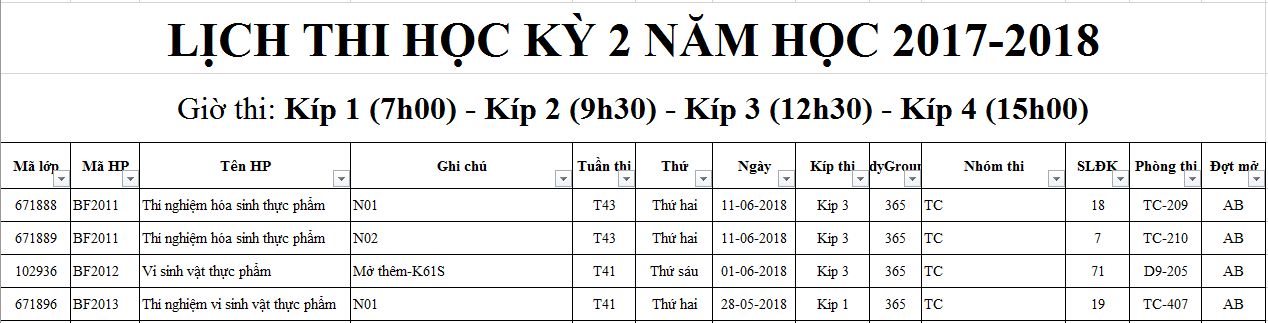


## Tại sao CSDL lại không có liên kết?

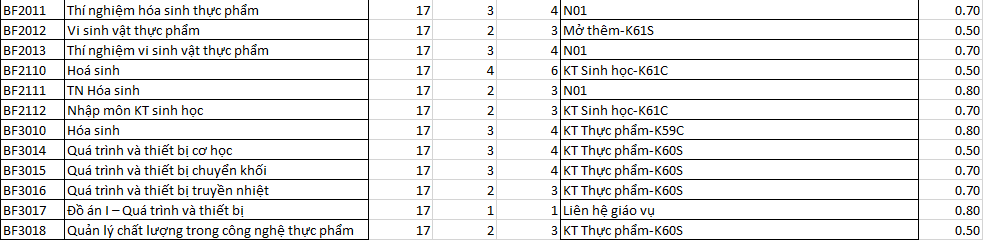
Sau khi tham khảo ý kiến từ nhiều người có kinh nghiệm, em nhận thấy rằng khi các bảng trong CSDL có liên kết với nhau, thì khi thao tác ở 1 bảng sẽ phải tham chiếu đến bảng được liên kết, làm giảm tốc độ truy vấn. Khi các bảng không có liên kết với nhau việc truy vấn sẽ nhanh hơn, ngoài ra hoàn toàn có thể triển khai các bảng phân tán ở nhiều server khác nhau. Khi đó khi mà truy vấn đến 1 bảng bị lỗi, hoàn toàn không ảnh hưởng đến việc truy vấn của toàn bộ hệ thống. Việc thực hiện các phép JOIN, ràng buộc sẽ được thực hiện ở Server.

## Tìm kiếm dữ liệu mẫu

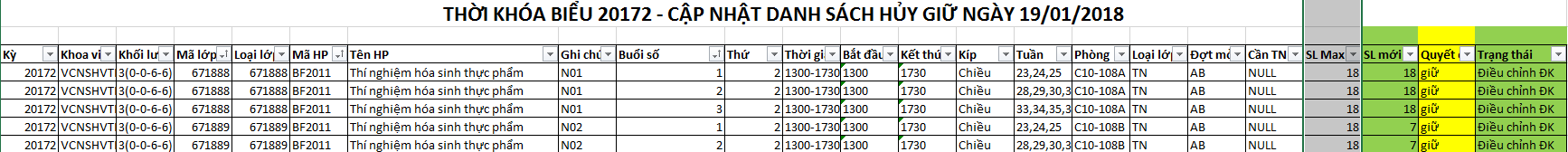
Đây là phần tốn rất nhiều công sức trong quá trình thực hiện đồ án. Một bộ dữ liệu về Sinh viên, Học phần, Lớp học đủ số lượng lại vô cùng khó. Tất cả các dữ liệu được về học phần, lớp học, mã môn học, sinh viên được lấy về từ lịch thi học kỳ, danh sách điểm danh... từ phòng đào tạo của trường ĐH Bách Khoa Hà Nội.



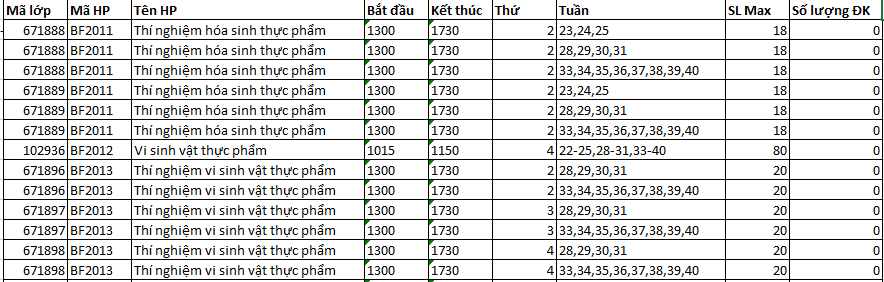
Nhưng chưa thể lấy luôn được bộ dữ liệu đó. Em đã phải dùng đến các kỹ thuật Excel để lấy dữ liệu từ các cột, chỉnh sửa lại đúng định dạng trong CSDL, điền thêm vào các dữ liệu bị thiếu.

Kết quả thu được là 1 bảng gần 2000 học phần, tương đối đầy đủ để thực nghiệm.

Dữ liệu về lớp học được lấy từ thời khóa biểu của phòng đào tạo:.



Sau đó lược bỏ các dữ liệu thừa, điền thêm các dữ liệu thiếu, định dạng lại dữ liệu để được bộ dữ liệu về danh sách lớp hoàn chính.



# Kết luận và hướng phát triển

## Kết luận

Sản phẩm đã giải quyết được một số vấn đề đặt ra, tuy nhiên vẫn chỉ ở mức độ lý thuyết và chưa có ứng dụng thực tế hoàn toàn. Còn rất nhiều vấn đề thiếu sót như Hệ thống Bảo mật yếu kém, nghiệp vụ chưa hoàn toàn sát với thực tế, giao diện chưa bắt mắt...

Nguyên nhân do trình độ của em còn hạn chế và một hệ thống nghiệp vụ như thế này cần sự làm việc của rất nhiều người trong một khoảng thời gian dài, với các quy trình đầy đủ và sự tham vấn của nhiều bên.

## Hướng phát triển

Để phát triển sản phẩm mang tính thực tiễn cao hơn và có thể áp dụng thực tế, trước hết cần phải bám sát nghiệp vụ hơn, làm sao cho các yêu cầu nghiệp vụ quản lý đặt ra phải được thực hiện đầy đủ, chính xác.

Tiếp sau đó là mô hình hệ thống cần được cái tiến để đạt hiệu suất cao hơn. Mô hình hiện tại vẫn còn một số nhược điểm như chưa phân hóa rõ nhiệm vụ của các tầng, còn sự trùng lặp phương thức, trùng lặp code.

Cuối cùng là các biện pháp bảo mật như token cho API, chống tấn công hệ thống,...

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | World Wide Web Consortium (W3C), "W3C," World Wide Web Consortium (W3C), [Online]. Available: https://www.w3.org/. [Accessed 25 05 2018]. |
| [2] | D. K. Taft, jQuery Eases JavaScript, AJAX Development, Eweek, 2006. |
| [3] | D. Wellman, jQuery UI 1.8: The User Interface Library for jQuery (1st ed.), ISBN 978-1849516525, 2011. |
| [4] | J. G. Andrew Stellman, Head First C#, 3rd Edition, O'Reilly Media, ISBN-13: 978-1449343507, 2013. |
| [5] | J. Richter, Applied Microsoft® .NET Framework Programming (Developer Reference), ISBN-13: 978-0735614222. |
| [6] | S. R. Leonard Richardson, RESTful Web Services, O'Reilly Media, ISBN-13: 978-0596529260, 2008. |
| [7] | B. Lakshmiraghavan, Practical ASP.NET Web API 1st ed. Edition, ISBN-13: 978-1430261759. |
| [8] | Microsoft, "docs.microsoft.com," Microsoft, 04 27 2010. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff649690(v=pandp.10). [Accessed 25 05 2018]. |
| [9] | [Online]. |