|  |  |
| --- | --- |
| DIỆP THỊ LINH | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC  NGÀNH CÔNG NGHỆ THÔNG TIN |
|  |
| **Xây dựng Website Quản lý sinh viên**  **sử dụng Spring Boot và ReactJs** |
|  |
|  |
| **CBHD: TS. Nguyễn Bá Nghiễn** |
| CÔNG NGHỆ THÔNG TIN | **Sinh viên: Diệp Thị Linh** |
| **Mã số sinh viên: 2020600872** |
|  |
|  |
|  |
| Hà Nội – Năm 2024 |
|  |

**MỤC LỤC**

[LỜI NÓI ĐẦU ii](#_Toc132764496)

[DANH MỤC CÁC CHỮ VIẾT TẮT iii](#_Toc132764497)

[DANH MỤC BẢNG BIỂU iv](#_Toc132764498)

[DANH MỤC HÌNH ẢNH v](#_Toc132764499)

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc132764500)

[1.1. Tên đề tài 1](#_Toc132764501)

[1.2. Lý do chọn đề tài 1](#_Toc132764502)

[1.3. Mục tiêu của đề tài 1](#_Toc132764503)

[1.4. Đối tượng và phạm vi 1](#_Toc132764504)

[1.5. Kết quả dự kiến đạt được 1](#_Toc132764505)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT VÀ CÔNG CỤ SỬ DỤNG 1](#_Toc132764506)

[2.1. Cơ sở lý thuyết 1](#_Toc132764507)

[2.2. Công cụ sử dụng 1](#_Toc132764508)

[CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 1](#_Toc132764509)

[3.1. Phân tích hệ thống 1](#_Toc132764510)

[3.2. Biểu đồ use case tổng quát 1](#_Toc132764511)

[3.3. Mô hình thực thể liên kết 1](#_Toc132764512)

[3.4. Đặc tả use case 1](#_Toc132764513)

[3.5. Thiết kế hệ thống 1](#_Toc132764514)

[KẾT LUẬN 1](#_Toc132764515)

[TÀI LIỆU THAM KHẢO 1](#_Toc132764516)

# 

# LỜI NÓI ĐẦU

Ngày nay, việc ứng dụng công nghệ thông tin vào quản lý giáo dục đã phát triển mạnh ở các nước tiên tiến trên thế giới. Đặc biệt với các trường đại học có hàng nghìn sinh viên thì việc áp dụng công nghệ thông tin vào quản lý sinh viên là điều vô cùng cần thiết.

Sau khi tìm hiểu và dưới sự hướng dẫn của thầy Nguyễn Bá Nghiễn, em đã phân tích và thiết kế được **Website Quản lý sinh viên** để giúp giáo viên, giảng viên, nhà trường có thể dễ dàng quản lý được thông tin của sinh viên từ thông tin cá nhân, điểm số, thành tích. Đó cũng là lý do mà em lựa chọn đề tài này.

Để hoàn thành được đồ án tốt nghiệp này, em xin được gửi lời cảm ơn chân thành đến các thầy cô trong khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội đã tận tình giảng dạy và trang bị kiến thức cho em trong suốt thời gian em học tập tại trường. Thầy giáo hướng dẫn đề tài – **Tiến sĩ Nguyễn Bá Nghiễn**, giảng viên Khoa Công nghệ Thông tin Trường Đại học Công Nghiệp Hà Nội – đã tận tụy hết lòng giúp đỡ, hướng dẫn, chỉ dẫn tận tình để giúp em hoàn thành được đồ án.

***Hà Nội, Ngày 04 tháng 05 năm 2024***

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| Từ viết tắt | Giải thích |
| JRE | Java Runtime Environment |
| API | Application Programming Interface |
| OOP | Object Oriented Programming |
| ANSI | American National Standards Institute |
| POSIX | Portable Operating System Interface |
| HTML | HyperText Markup Language |
| CSS | Cascading Style Sheets |
| JSF | JavaServer Faces |
| IBM | International Business Machines |
| AOP | Aspect Oriented Programming |
| JDBC | Java Database Connectivity |
| POJO | Plain Old Java Object |
| JMS | Java Message Service |
| JTA | Java Transaction API |
| JMX | Java Management Extensions |
| RMI | Remote Method Invocation |
| MVC | Model-View-Controller |
| UI | User Interface |
| GUI | Graphical User Interface |
| AMQP | Advanced Message Queuing Protocol |
| XML | Extensible Markup Language |
| MVCC | Multiversion concurrency control |
| PK | Primary key |
| PFK | Primary Foreign Key |

# 

# DANH MỤC BẢNG BIỂU

# DANH MỤC HÌNH ẢNH

# TỔNG QUAN VỀ ĐỀ TÀI

## Tên đề tài

***Website Quản lý sinh viên sử dụng Spring Boot và ReactJs***

## Lý do chọn đề tài

Hiện nay các trường đại học tại Việt Nam cũng như các doanh nghiệp đang trong quá trình ứng dụng sự phát triển mạnh mẽ của công nghệ thông tin vào quản lý con người. Việc ứng dụng công nghệ vào việc quản lý con người giúp cho việc quản lý được dễ dàng hơn, thông tin có độ chính xác cao hơn, tiết kiệm thời gian hơn. Nhất là việc ứng dụng công nghệ đối với các trường đại học có hàng ngàn sinh viên đang theo học càng bức thiết hơn.

Đối với hệ thống quản lý sinh viên khi chưa ứng dụng công nghệ thông tin tồn tại rất nhiều khuyết điểm như:

* Tốn nhiều thời gian vào việc tìm kiếm, phân loại và sữa chữa thông tin.
* Tốn nhiều không gian lưu trữ, dữ liệu dễ dàng bị mất và khó khôi phục.
* Tốn nhiều chi phí, nhân lực vào việc quản lý.
* Việc kiểm tra theo dõi còn gặp nhiều khó khăn.

Đối với hệ thống quản lý sinh viên khi ứng dụng công nghệ thông tin:

* Ưu điểm:
  + Tiết kiệm được nhiều thời gian và chi phí.
  + Quản lý hiệu quả.
  + Dễ dàng kiểm tra theo dõi.
  + Xử lý được dữ liệu lơn, chính xác, nhanh gọn.
* Nhược điểm:
  + Phụ thuộc vào internet

Thiết kế website là nhu cầu cần thiết đối với các trường đại học.

Nhận thức được sự quan trọng và cũng như để có thể áp dụng được những kiến thức đã được học và tìm hiểu, em xin được áp dụng những kiến thức đã được học và tìm hiểu đó để xây dựng một website cụ thể là: ***“*Xây dựng Website Quản lý sinh viên sử dụng Spring Boot và ReactJs*”.***

## Mục tiêu của đề tài

Đề tài: ***Website LMS quản lý sinh viên*** đáp ứng được những mục tiêu:

* Hiểu biết những kiến thức cần thiết về xây dựng giao diện trang web, phân tích thiết kế hệ thống, cơ sở dữ liệu.
* Nắm được các kiến thức cơ bản về Java, Spring Boot, JavaScript, ReactJS.
* Giúp cho quá trình quản lý sinh viên dễ dàng hơn, tiết kiệm thời gian hơn.

## Đối tượng và phạm vi

Giáo viên, giảng viên của trường đại học

## Kết quả dự kiến đạt được

Website bao gồm những mục chính và chức năng chính:

* Tạo tài khoản.
* Đăng nhập, đăng xuất, đổi mật khẩu.
* Quản lý sinh viên (Thêm sinh viên, Sửa thông tin sinh viên, Xem thông tin, điểm số của sinh viên tương ứng với các môn học, Thêm sinh viên từ file Excel, Biểu đồ thống kê điểm tích lũy của sinh viên).
* Nhập điểm ( Nhập điểm bằng file Excel, Sửa, Xóa điểm của sinh viên).
* Quản lý trạng thái các file đã upload.
* Upload, Download file.
* Quản lý môn học, Quản lý lớp học, Thêm sinh viên vào lớp học.

# CƠ SỞ LÝ THUYẾT VÀ CÔNG CỤ SỬ DỤNG

## Cơ sở lý thuyết

### Ngôn ngữ UML

UML là viết tắt của “Unified Modeling Language” - là ngôn ngữ mô hình hóa thống nhất, dùng để đặc tả, hình dung, xây dựng và tư liệu hóa phần mềm hướng đối tượng.

UML dùng để:

* Trực quan hoá: Đưa ra các mô hình trực quan về hệ thống để mọi người có thể hiểu được.
* Đặc tả: UML cho phép xây dựng các mô hình đúng đắn và đầy đủ để mô tả hệ thống.
* Xây dựng: Các mô hình UML có thể liên kết trực tiếp với nhiều ngôn ngữ lập trình Java, C++, VB, ...
* Lập tài liệu: UML cho phép tạo các tài liệu như tài liệu về các yêu cầu của người dùng, kiến trúc hệ thống, thiết kế hệ thống, kiểm thử, kế hoạch dự án, ...

UML có thể được sử dụng làm công cụ giao tiếp giữa người dùng, nhà phân tích, nhà thiết kế và nhà phát triển phần mềm.

Các phần tử của UML:

View (Quan sát): Theo các phương diện khác nhau của hệ thống cần phân tích, thiết kế. Dựa vào các quan sát để thiết lập kiến trúc cho hệ thống cần phát triển.

***Có 9 loại biểu đồ:***

* Use Case Diagram (Biểu đồ ca sử dụng): Mô tả sự tương tác giữa các tác nhân và hệ thống thông qua các ca sử dụng.
* Class Diagram (Biểu đồ lớp): Biểu đồ lớp chỉ ra sự tồn tại giữa các lớp và mối quan hệ giữa chúng trong thiết kế logic trong hệ thống. Khung nhìn tĩnh của hệ thống chủ yếu hỗ trợ các chức năng của hệ thống.
* Object Diagram (Biểu đồ đối tượng): Biểu đồ đối tượng đại diện cho một thể hiện của biểu đồ lớp, cho nên các khái niệm cơ bản là giống biểu đồ lớp. Biểu đồ đối tượng được sử dụng để hiện thị một tập hợp các đối tượng và các mối quan hệ của chúng như một thể hiện.
* Sequence Diagram (Biểu đồ trình tự): Là biểu đồ để xác định và chỉ rõ vai trò của đối tượng tham gia vào luồng sự kiện của use case.
* Collaboration Diagram (Biểu đồ cộng tác): Tương tự như biểu đồ trình tự nhưng nhấn mạnh vào sự tương tác của các đối tượng trên cơ sở cộng tác với nhau bằng cách trao đổi các thông điệp để thực hiện các yêu cầu theo ngữ cảnh công việc.
* State Diagram (Biểu đồ trạng thái): Thể hiện chu kỳ hoạt động của các đối tượng, của các hệ thống con và của cả hệ thống.
* Activity Diagram (Biểu đồ hành động): Được sử dụng để mô tả các hoạt động và các hành động được thực hiện trong một use case.
* Component Diagram (Biểu đồ thành phần): Chỉ ra cấu trúc vật lý của các thành phần trong hệ thống.
* Deployment Diagram (Biểu đồ triển khai): Chỉ ra cách bố trí vật lý các thành phần theo kiến trúc được thiết kế của hệ thống.

Relationship (Quan hệ):

* Realization (Hiện thực hoá):Là quan hệ ngữ nghĩa giữa giao diện và lớp.
* Dependency (Phụ thuộc):Là quan hệ ngữ nghĩa giữa hai phần tử, trong đó sự thay đổi của một tử sẽ tác động đến ngữ nghĩa của phần tử phụ thuộc.
* Generalization (Kế thừa): Là quan hệ mô tả sự khái quát hoá mà trong đó một số đối tượng cụ thể (của lớp con) sẽ được kế thừa các thuộc tính, các phương thức của các đối tượng tổng quát (lớp cơ sở).
* Association (Kết hợp): Là quan hệ cấu trúc xác định mối liên kết giữa các lớp đối tượng.

### Java logo and symbol, meaning, history, PNGNgôn ngữ Java

Hình 2.1. Java logo

**Java** là một ***Ngôn ngữ lập trình*** và là một ***Platform***.

Ngôn ngữ lập trình: Java là một ngôn ngữ lập trình bậc cao, hướng đối tượng, bảo mật và mạnh mẽ.

Platform: Bất cứ môi trường phần cứng hoặc phần mềm nào mà trong đó có một chương trình chạy, thì được hiểu như là một Platform. Với môi trường runtime riêng cho mình và API, Java được gọi là Platform.

Ngôn ngữ lập trình Java ban đầu được phát triển bởi Sun Microsystems do James Gosling khởi xướng và được phát hành vào năm 1995. Phiên bản mới nhất của Java Standard Edition được biết đến thời điểm hiện tại là Java SE 18. Với sự tiến bộ của Java và sự phổ biến rộng rãi của nó, nhiều cấu hình đã được xây dựng để phù hợp với nhiều loại nền tảng khác nhau. Ví dụ: J2EE cho các ứng dụng doanh nghiệp, J2ME cho các ứng dụng di động.

Các phiên bản J2 mới đã được đổi tên thành Java SE, Java EE và Java ME. Phương châm của java là “Write Once, Run Anywhere” – viết một lần chạy nhiều nơi, nghĩa là chỉ cần viết một lần trên window chẳng hạn, sau đó vẫn chương trình đó có thể chạy trên Linux, Android, các thiết bị J2ME…

**Các đặc điểm của Java:**

**Hướng đối tượng** – Trong Java, mọi thứ đều là một Object. Java có thể dễ dàng mở rộng và bảo trì vì nó được xây dựng dựa trên mô hình Object.

**Độc lập nền tảng** – Không giống nhiều ngôn ngữ lập trình khác bao gồm cả C và C ++, khi Java được biên dịch, nó không được biên dịch thành ngôn ngữ máy nền tảng cụ thể, thay vào đó là những mã bytecode nền tảng độc lập. Mã byte này được thông dịch bởi máy ảo trên nền tảng nào đó mà nó đang chạy.

**Đơn giản** – Java được thiết kế để dễ học. Chỉ cần hiểu khái niệm cơ bản về OOP Java, thì sẽ rất dễ để trở thành master về Java.

**Bảo mật** – Với các thuật toán mã hoá của Java, nó cho phép phát triển các hệ thống không có virus, giả mạo, bảo mật rất tốt.

**Kiến trúc – trung lập** – Trình biên dịch Java tạo ra định dạng tệp đối tượng kiến trúc trung lập, làm cho mã biên dịch được thực thi trên nhiều bộ vi xử lý, với sự hiện diện của nền tảng Java.

**Portable** – Là kiến trúc tập trung và không có khía cạnh thực hiện phụ thuộc của đặc tả này làm cho Java khả chuyển. Trình biên dịch trong Java được viết bằng ANSI C, đó là một tập con POSIX.

**Mạnh mẽ** – Java làm nỗ lực để loại trừ các tình huống dễ bị lỗi bằng cách kiểm tra lỗi tại thời gian biên dịch và kiểm tra lỗi tại runtime.

**Đa luồng** – Với tính năng đa luồng của Java có thể viết các chương trình có thể thực hiện nhiều tác vụ đồng thời. Tính năng thiết kế này cho phép các nhà phát triển xây dựng các ứng dụng tương tác có thể chạy trơn tru hơn.

**Thông dịch** – Java là ngôn ngữ lập trình vừa có biên dịch vừa có thông dịch. Mã byte Java được dịch trực tiếp tới các máy tính gốc và không được lưu trữ ở bất cứ đâu.

**Hiệu năng cao** – Với việc sử dụng trình biên dịch Just-In-Time, Java cho phép thực hiện hiệu năng cao, nhanh chóng phát hiện và gỡ lỗi. Hơn nữa quá trình thu gom rác garbage collection tìm duyệt các đối tượng không sử dụng và xóa chúng để giải phóng bộ nhớ.

**Phân tán** – Java được thiết kế cho môi trường phân tán của Internet.

**Linh động** – Java được xem là linh hoạt hơn C hoặc C++ vì nó được thiết kế để thích nghi với môi trường đang phát triển. Các chương trình Java có thể mang một lượng lớn thông tin tại runtime mà có thể được sử dụng để xác minh và giải quyết các truy cập vào các đối tượng tại runtime.

### File:Unofficial JavaScript logo 2.svg - Wikimedia CommonsNgôn ngữ JavaScript

Hình 2.2. JavaScript logo

JavaScript là một ngôn ngữ lập trình đa nền tảng (cross-platform), ngôn ngữ lập trình kịch bản, hướng đối tượng. JavaScript là một ngôn ngữ nhỏ và nhẹ (small and lightweight). JavaScript chứa các thư viện tiêu chuẩn cho các đối tượng (object), ví dụ như: Array, Date, Math, và các yếu tố cốt lõi của ngôn ngữ lập trình như: toán tử (operators), cấu trúc điều khiển (control structures), và câu lệnh. JavaScript có thể được mở rộng cho nhiều mục đích bằng việc bổ sung thêm các object; ví dụ:

* Client-side JavaScript - JavaScript phía máy khách, JavaScript được mở rộng bằng cách cung cấp các object để quản lý trình duyệt và Document Object Model (DOM) của nó. Ví dụ, phần mở rộng phía máy khách cho phép một ứng dụng tác động tới các yếu tố trên một trang HTML và phản hồi giống các tác động của người dùng như click chuột, nhập form, và chuyển trang.
* Server-side JavaScript - JavaScript phía máy chủ, JavaScript được mở rộng bằng cách cung cấp thêm các đối tượng cần thiết để để chạy JavaScript trên máy chủ. Ví dụ, phần mở rộng phía server này cho phép ứng dụng kết nối với cơ sở dữ liệu (database), cung cấp thông tin một cách liên tục từ một yêu cầu tới phần khác của ứng dụng, hoặc thực hiện thao tác với các tập tin trên máy chủ.

JavaScript được phát triển bởi Brendan Eich, một nhân viên của Netscape, vào tháng 9 năm 1995. Được biết đến lần đầu tiên là Mocha, rồi sau đó được đổi tên thành LiveScript và cuối cùng là JavaScript nổi tiếng như hiện giờ. Phiên bản đầu tiên của ngôn ngữ này bị giới hạn độc quyền bởi Netscape và chỉ có các tính năng hạn chế, nhưng nó tiếp tục phát triển theo thời gian.

Năm 1996, JavaScript được chính thức đặt tên là ECMAScript. ECMAScript 2 phát hành năm 1998 và ECMAScript 3 tiếp tục ra mắt vào năm 1999. Nó liên tục phát triển thành JavaScript ngày nay, giờ đã hoạt động trên khắp mọi trình duyệt và trên khắp các thiết bị từ di động đến máy tính bàn.

JavaScript liên tục phát triển kể từ đó, có lúc đạt đến 92% website đang sử dụng JavaScript vào năm 2016. Chỉ trong 20 năm, nó từ một ngôn ngữ lập trình riêng trở thành công cụ quan trọng nhất trên bộ công cụ của các chuyên viên lập trình web.

JavaScript cùng với HTML, CSS đã trở thành ngôn ngữ không thể thiếu trong lập trình web hiện nay:

* Với việc phát triển mạnh mẽ của công nghệ, JavaScript đã dần trở thành một ngôn ngữ đa nền tảng với nhiều ứng dụng khác nhau.
* Các Framework, Library cho lập trình front-end mạnh mẽ hỗ trợ xây dựng các trang Single-page application (SPA) như: ReactJS, Angular, VueJS.
* Hỗ trợ lập trình mobile bằng thư viện React Native.
* Hỗ trợ lập trình Window bằng Electron.
* Không chỉ phát triển ở khía cạnh lập trình phía client-side, JavaScript cũng hỗ trợ viết server-side rất mạnh mẽ với NodeJS.

Với hệ sinh thái mã nguồn mở khổng lồ cùng sự phát triển như hiện nay, JavaScript được dự đoán sẽ phát triển mạnh mẽ hơn nữa trong những năm tới.

### Spring Framework

#### Spring Framework là gì?

Framework là các đoạn code đã được viết sẵn, cấu thành nên một bộ khung và các thư viện lập trình được đóng gói. Chúng cung cấp các tính năng có sẵn như mô hình, API và các yếu tố khác để tối giản cho việc phát triển các ứng dụng web phong phú, năng động.

Spring Framework là một Framework có vai trò phát triển cho các ứng dụng Java. Trong số đó, phổ biến nhất là Java Enterprise và nó được sử dụng bởi hàng triệu lập trình viên khác nhau.

Spring được coi là một khung framework cho mọi framework vì trong Spring Framework lại cung cấp nhiều framework như Struts, Hibernate, EJB, JSF.

#### Lịch sử phát triển của Spring Framework

EJB (Enterprise JavaBeans) là một kiến trúc phát triển dành cho các ứng dụng doanh nghiệp đòi hỏi tính mạnh mẽ và khả năng mở rộng cao. Nó là một thành phần quan trọng của nền tảng Java EE. Nó được tạo ra bởi hai gã khổng lồ công nghệ là IBM và Sun vào những năm cuối thập niên 90. EJB là một framework lớn với độ phức tạp cao. Tuy được phát triển để dễ sử dụng hơn nhưng chưa đáp ứng được mục tiêu ban đầu tạo ra nó.

Năm 2002, Spring Framework phát hành phiên bản đầu tiên bởi Rod Johnson. Việc xây dựng các ứng dụng doanh nghiệp trở nên đơn giản và dễ dàng hơn. Điều này đã cho Spring Framework rất nhiều động lực để thay thế EJB. Theo đà phát triển đó, Spring đã trở thành framework mã nguồn mở phổ biến nhất để xây dựng các ứng dụng doanh nghiệp. Theo một số nguồn, trên 50% các ứng dụng web Java hiện nay đang sử dụng Spring.

#### Mục tiêu của Spring Framework

* Làm cho công việc phát triển trở lên đơn giản hơn với Java.
* Nới lỏng ràng buộc thành phần thông qua Dependency Injection.
* Lập trình hướng khía cạnh (AOP), điều này cho phép thêm một số services trên toàn ứng dụng cho đối tượng.
* Giảm thiểu các mã boilerplate thông qua việc sử dụng template và aspect.

#### Ưu điểm khi sử dụng Spring Framework

* Xác định template trước: Spring Framework cung cấp những template cho Hibernate, Struts, JDBC, v.v. Vì vậy, lập trình viên không cần phải viết quá nhiều code và có thể bỏ qua những bước cơ bản.
* Dễ dàng kiểm thử: Dependency Injection làm cho việc kiểm thử phần mềm, ứng dụng được tiến hành dễ dàng hơn. Nếu như ứng dụng EJB yêu cầu máy chủ phải chạy lại ứng dụng thì Spring Framework thì không.
* Dung lượng nhỏ: Spring Framework có dung lượng nhỏ nhờ vào mô hình POJO. Spring Framework không bắt lập trình viên phải kế thừa các lớp hay thực thi qua bất kì giao diện nào. Đó là lý do Spring Framework được coi là bất khả xâm phạm.
* Phát triển nhanh: Dependency Injection tạo nên tính năng đặc biệt của Spring Framework giúp hỗ trợ các framework khác và giúp cho việc phát triển ứng dụng Java EE dễ dàng hơn.
* Hỗ trợ khai báo: Spring Framework cung cấp những hỗ trợ khai báo về bộ nhớ đệm, xác nhận, giao dịch và định dạng.
* Tính trừu tượng: Nó cung cấp sự trừu tượng hóa cao cho các chi tiết Java EE như JMS, JDBC, JPA và JTA.

#### Kiến trúc của Spring Framework

Hình 2.3. Kiến trúc của Spring Framework

* Test: Tầng này cung cấp khả năng hỗ trợ kiểm thử với JUnit và TestNG.
* Core Container là thành phần cốt lõi không thể thiếu của Spring. Về cơ bản nó đóng vai trò như là một nhà máy sản xuất các Beans
* Spring Bean là trung tâm của Core và là trái tim của một ứng dụng Spring. Trái ngược với EJB, Spring Framework được thiết kế từ lõi bằng cách sử dụng các POJO hay các Spring Bean. Spring Bean có thể được hiểu là các đối tượng Java đơn giản. Điều này khá tương đồng với nguyên lý thiết kế “Đơn trách nhiệm” (single responsibility) của Robert C Martin trong lập trình hướng đối tượng.
* Spring Context mang mọi thứ lại với nhau. Spring Context kế thừa các tính năng của Spring Bean và bổ sung các hỗ trợ cho internationalization (ví dụ như các resource bundle), event propagation, resource loading… Ngoài ra, Spring Context cũng hỗ trợ các tính năng của Java EE như EJB, JMX và RMI. Interface ApplicationContext chính là trọng tâm của Spring Context.
* Spring Expression Language là một ngôn ngữ ngắn gọn giúp cho việc cấu hình Spring Framework trở nên linh hoạt hơn.
* AOP Section (AOP, Aspects, Instrumentation, messaging) là nơi cho phép lập trình hướng khía cạnh (Aspect Oriented Programming). Nói một cách ngắn gọn, AOP cho phép tạo ra các services cho toàn ứng dụng. Ví dụ: ghi log, bảo mật, transaction, … sau đó có thể áp dụng các services này cho đối tượng khác mà không cần phải sửa đổi code bên trong đối tượng đó
* Data access layer là nơi giao tiếp với cơ sở dữ liệu. Ở đây có JDBC, Spring cung cấp một số class trợ giúp (JDBC Helper) làm cho việc truy cập Database trở lên dễ dàng hơn. Bằng cách sử dụng các Spring JDBC helper, có thể giảm hơn 50% code khi giao tiếp với Database.
* ORM (Object Relational Mapping) là một thành phần phổ biến của module này, cho phép ánh xạ các đối tượng Java với các bảng trong Database, làm cho việc lập trình với cơ sở dữ liệu trở lên gần gũi với thực tế hơn.
* Spring hỗ trợ quản lý các giao dịch (Transactions), có thể sử dụng giao dịch trên các phương thức, khi gọi database, hoặc bất cứ điều gì. Nó rất linh hoạt. Rất nhiều dự án sử dụng trình quản lý giao dich (Transaction Manager) của Spring.
* Web layer chứa tất cả các class liên quan tới Web. Có thể nói đây là ngôi nhà của Spring MVC.

#### Các dự án của Spring Framework

* Spring MVC: thiết kế dành cho việc xây dựng các ứng dụng nền tảng web.
* Spring Data: cung cấp một cách tiếp cận đúng đắn để truy cập dữ liệu từ cơ sở dữ liệu quan hệ, phi quan hệ, MapReduce và thậm chí còn hơn thế nữa.
* Spring Security: Dự án này cung cấp các cơ chế xác thực (authentication) và phân quyền (authorization) cho ứng dụng.
* Spring Boot: là một framework giúp chúng ta phát triển cũng như chạy ứng dụng một cách nhanh chóng.
* Spring Batch: dự án này giúp chúng ta dễ dàng tạo các lịch trình (scheduling) và tiến trình (processing) cho các công việc xử lý theo mẻ (batch job).
* Spring Integration: là một implementation của Enterprise Integration Patterns (EIP). Dự án này thiết kế một kiến trúc hướng thông điệp hỗ trợ việc tích hợp các hệ thống bên ngoài.
* Spring Social: dự án này sẽ kết nối ứng dụng với các API bên thứ ba như Facebook, Twitter, ...

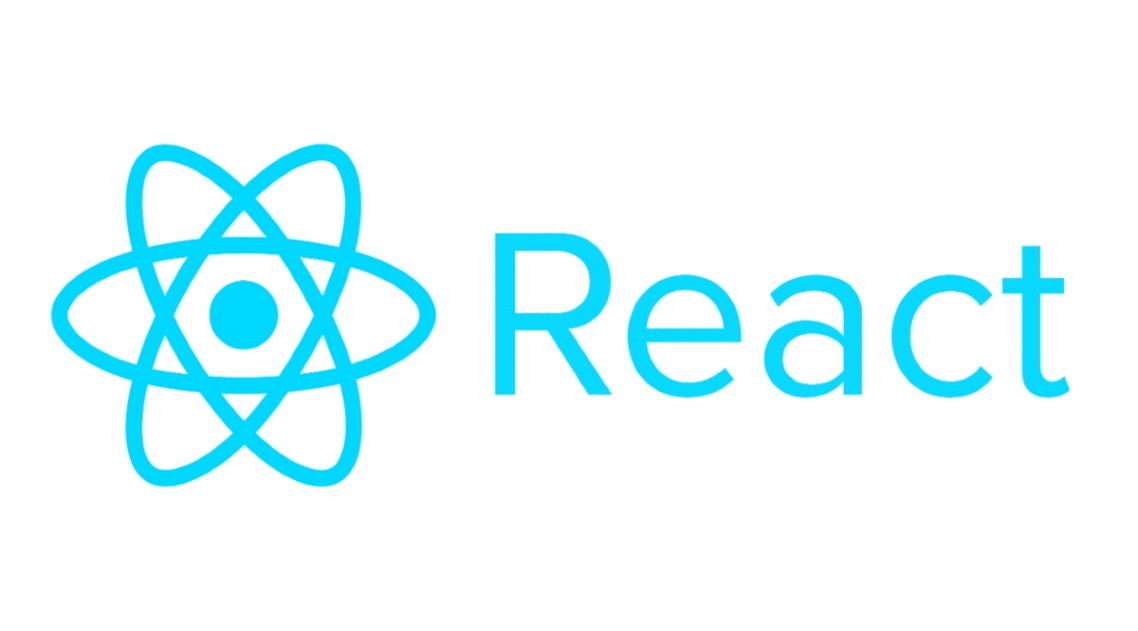
Ngoài các dự án lớn kể trên, còn có các dự án **Spring Cloud, Spring Session, Spring AMQP, Spring for Apache Kafka, Spring Web Services, ...**

### ReactJS

#### ReactJS là gì?

ReactJS là một thư viện JavaScript mã nguồn mở dùng để xây dựng giao diện người dùng hoặc các thành phần UI.

ReactJS là một thư viện JavaScript với xu hướng Single Page Application.



Hình 2.4. React logo

Hiện tại, nó đã đi trước các đối thủ chính như Angular và [Bootstrap](https://getbootstrap.com/). Được đánh giá cao và trở nên rất phổ biến với hơn 1.300 developer và hơn 94.000 trang web đang sử dụng.

#### Lịch sử phát triển của ReactJS

Trước thời điểm năm 2013 việc xây dựng UI cho Facebook hoàn toàn dựa trên framework AngularJS của Google, tuy nhiên sau thời điểm năm 2013 khi lượng người dùng trở nên quá lớn và dữ liệu Facebook lưu trữ ngày càng nhiều, kéo theo đó việc xử lý dữ liệu ở ngoài UI trở nên quá khó khăn vì AngularJS chậm và khá nặng nề. Facebook không thể tìm kiếm được một framework nào khác đủ khả năng thay thế cho AngularJS, nên đã quyết định tự tạo ra ra một công nghệ mới là “React” gồm 2 phần: ReactJS (dùng cho web app) và React Native (dùng cho mobile app).

React được tạo bởi Jordan Walke, một kỹ sư phần mềm làm việc tại Facebook, người đã tại ra bản mẫu đầu tiền của ReactJS gọi là “FaxJS”. Jordan Walke bị ảnh hưởng bới XHP (một nền tảng thành phần HTML cho PHP). React lần đầu tiên được triển khai cho ứng dụng Newsfeed của Facebook năm 2011, rồi sau đó được triển khai cho Instagram vào năm 2012. Tháng 5 năm 2013, nó được mở mã nguồn (open-sourced) tại JSConfUS nhằm mục đích có nhiều người góp phần phát triển thư viện này và sớm trở thành một framework.

Ngày 26/04/2022, React đã cho ra phiên bản mới nhất là phiên bản 18.1.0.

#### Tính khả dụng

* Tạo ra những Website chạy thật mượt mà, nhanh, có khả năng mở rộng cao và đơn giản thực hiện.
* Cho phép phá vỡ giao diện người dùng, biến nó từ phức tạp thành đơn giản hơn.

#### Các đặc trưng của ReactJS

* Component: là thành phần cơ bản của ReactJS. Nó là một đoạn mã ngắn có ý nghĩa thể hiện một chức năng hay đối tượng nhất định, có thể tái sử dụng nhiều lần với các thuộc tính, trạng thái khác nhau. Mọi thứ trong ReactJS đều là “component” và giúp bảo trì code khi làm việc với dự án lớn.
* Virtual DOM: công nghệ DOM ảo giúp tăng hiệu năng cho ứng dụng. Việc chỉ node gốc mới có trạng thái và khi nó thay đổi sẽ tái cấu trúc lại toàn bộ, đồng nghĩa với việc DOM tree cũng sẽ phải thay đổi một phần, điều này sẽ ảnh hưởng đến tốc độ xử lý. Virtual DOM là một object JavaScript, mỗi object chứa đầy đủ thông tin cần thiết để tạo ra một DOM, khi dữ liệu thay đổi nó sẽ tính toán sự thay đổi giữa object và tree thật, điều này sẽ giúp tối ưu hoá việc re-render DOM tree thật.
* JSX: là thành phần quan trọng trong ReactJS. JSX là một dạng ngôn ngữ cho phép viết các mã HTML trong JavaScript. Đây là cú pháp mở rộng JavaScript, là sự kết hợp của XML và JavaScript, giúp chuyển đổi cú pháp dạng XML thành dạng JavaScript, sử dụng cú pháp của XML để code. Đặc điểm khi sử dụng JSX là nhanh hơn, an toàn hơn và dễ sử dụng hơn.
* Props và State: props, viết tắt của properties, là các thuộc tính của một component. Nếu component là một đối tượng thì props là những tính chất, đặc trưng của một đối tượng. Props là một giá trị bất biến. State được dùng để biểu diễn trạng thái của component. Tức là giá trị của state có thể thay đổi được. Mỗi khi state của một component thay đổi thì component đồng thời render lại để cập nhật UI.
* Redux: là một thư viện giúp quản lý các state một cách tốt hơn. Thay vì phải truyền state qua từng component, redux tạo ra một store duy nhất dùng để thay đổi dữ liệu.
* Luồng dữ liệu một chiều (one-way data binding): dữ liệu được truyền từ cha đến con thông qua props. Luồng dữ liệu đơn giản giúp chúng ta dễ dàng kiểm soát cũng như sửa lỗi. Thích hợp xây dựng ứng dụng lớn mà dữ liệu của chúng thay đổi liên tục theo thời gian. Dữ liệu thay đổi thì hầu hết kèm theo sự thay đổi về giao diện.

#### Lợi ích khi sử dụng ReactJS

* Dễ sử dụng: ReactJS hoạt động dựa trên cơ chế kết hợp giữa [HTML](https://bizfly.vn/techblog/html-la-gi.html) và JavaScript nên khá đơn giản và dễ sử dụng. ReactJS là một thư viện GUI mã nguồn mở JavaScript tập trung vào một điều cụ thể để hoàn thành nhiệm vụ UI hiệu quả.
* Hỗ trợ Reusable Component: ReactJS mang lại khả năng tái sử dụng các components đã phát triển thành những ứng dụng khác với chức năng không đổi.
* Viết component dễ dàng hơn: component dễ viết hơn vì nó sử dụng JSX vì JSX hiệu quả trong việc phát triển components đặc biệt hoặc các ứng dụng có khối lượng lớn, giúp làm rõ toàn bộ quá trình viết cấu trúc trang web. Component có thể mở rộng cú pháp tùy chọn cho JavaScript, cho phép kết hợp HTML với JavaScript.
* Hiệu suất tốt hơn với Virtual DOM: React sẽ cập nhật hiệu quả quá trình DOM (Document Object Model – Mô hình đối tượng tài liệu) sử dụng Virtual DOM. React cập nhật quá trình DOM và tự tạo cho nó các Virtual DOM, sau đó đưa chúng vào bộ nhớ. Hoạt động này thúc đẩy việc tự động cập nhật tự động của Virtual DOM ngay khi DOM thực tế có sự thay đổi, giúp tốc độ của các ứng dụng không bị gián đoạn.
* Thân thiện với SEO: ReactJS cho phép bạn tạo giao diện người dùng có thể được truy cập trên các công cụ tìm kiếm khác nhau.

### Cơ sở dữ liệu MySQL

#### MySQL là gì?



Hình 2.5. MySQL logo

MySQL là hệ quản trị cơ sở dữ liệu mã nguồn mở phổ biến hàng đầu trên thế giới và đặc biệt được ưa chuộng trong quá trình xây dựng, phát triển ứng dụng. Đây là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có khả năng thay đổi mô hình sử dụng phù hợp với điều kiện công việc khả chuyển. MySQL hoạt động trên nhiều hệ điều hành, cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.

Với tốc độ và tính bảo mật cao, MySQL thích hợp với các ứng dụng có truy cập cơ sở dữ liệu trên internet. MySQL có thể tải miễn phí từ trang chủ với nhiều phiên bản cho các hệ điều hành khác nhau như: phiên bản Win32 cho các hệ điều hành dòng Windows, Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS...

#### Vì sao sử dụng MySQL?

Các tính năng nổi bật trong MySQL:

* Sử dụng dễ dàng: MySQL là cơ sở dữ liệu tốc độ cao và ổn định, công cụ này dễ sử dụng và hoạt động trên nhiều hệ điều hành cung cấp hệ thống lớn các hàm tiện ích.
* Tính bảo mật cao: MySQL phù hợp với các ứng dụng có truy cập cơ sở dữ liệu trên internet vì nó sở hữu nhiều tính năng bảo mật, thậm chí là bảo mật cấp cao.
* Đa tính năng: MySQL có thể hỗ trợ hàng loạt các chức năng SQL từ hệ quản trị cơ sở dữ liệu quan hệ trực tiếp và cả gián tiếp.
* Khả năng mở rộng và mạnh mẽ: Công cụ MySQL có khả năng xử lý khối dữ liệu lớn và có thể mở rộng khi cần thiết.
* Tương thích trên nhiều hệ điều hành: MySQL tương thích để chạy trên nhiều hệ điều hành, như Novell NetWare, Windows \* Linux \*, nhiều loại UNIX \* (như Sun \* Solaris \*, AIX và DEC \* UNIX), OS / 2, FreeBSD \*,.... MySQL cũng cung cấp phương tiện mà các máy khách có thể chạy trên cùng một máy tính với máy chủ hoặc trên một máy tính khác (giao tiếp qua mạng cục bộ hoặc Internet).
* Cho phép khôi phục: MySQL cho phép các transactionđược khôi phục, cam kết và phục hồi sự cố.

Sử dụng MySQL cho dự án vì nó hỗ trợ các cấu trúc phức tạp và rất nhiều các kiểu dữ liệu được xây dựng sẵn và do người dùng định nghĩa. Nó cung cấp khả năng mở rộng dữ liệu, toàn vẹn dữ liệu và rất nhiều tính năng tiên tiến khác hỗ trợ cho việc xây dựng và cập nhật ứng dụng lâu dài.

Link tải phần mềm: <https://www.mysql.com/downloads/>

## Công cụ sử dụng

* IntelliJ IDEA: phục vụ mục đích code Java.
* Visual Studio Code: phục vụ mục đích code ReactJS.
* DBeaver: thao tác với cơ sở dữ liệu.
* Postman: sử dụng để thử nghiệm API.
* Chrome DevTools: gỡ lỗi khi phát triển web.
* Case Studio: thiết kế mô hình thực thể liên kết.
* Rational Rose: thiết kế hệ thống.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Phân tích hệ thống

### Vai trò của người dùng

* Hệ thống chia làm 2 nhóm người dùng chính:
  + Sinh viên
  + Giảng viên
  + Người quản trị
* Mô tả nhóm người dùng.
  + Sinh viên: Xem thông tin cá nhân, đăng ký học phần ...
* Giảng viên: Có thể quản lý thông tin sinh viên, upload, download file, nhập điểm ...
* Người quản trị: Quản lý user, Quản lý lớp học, Quản lý môn học ...

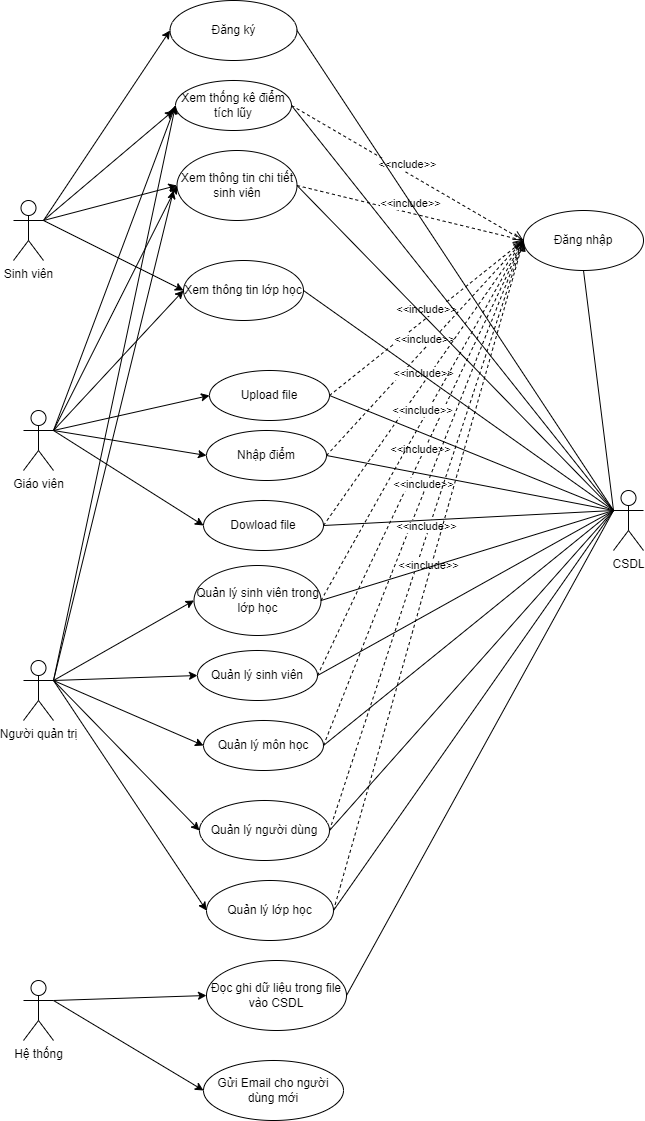
### Yêu cầu chức năng

* Chức năng chung:
  + Đăng nhập
  + Xem thông tin sinh viên
  + Đổi mật khẩu
* Sinh viên:
  + Đăng ký
  + Đăng ký học phần
  + Sửa thông tin sinh viên
* Giảng viên:
  + Nhập điểm
  + Upload file
  + Download file
  + Xem thông tin lớp học
* Quản trị viên:
  + Quản lý môn học
  + Quản lý người dùng
  + Quản lý lớp học
* Hệ thống:
  + Đọc ghi dữ liệu trong file
  + Gửi email

### Yêu cầu phi chức năng

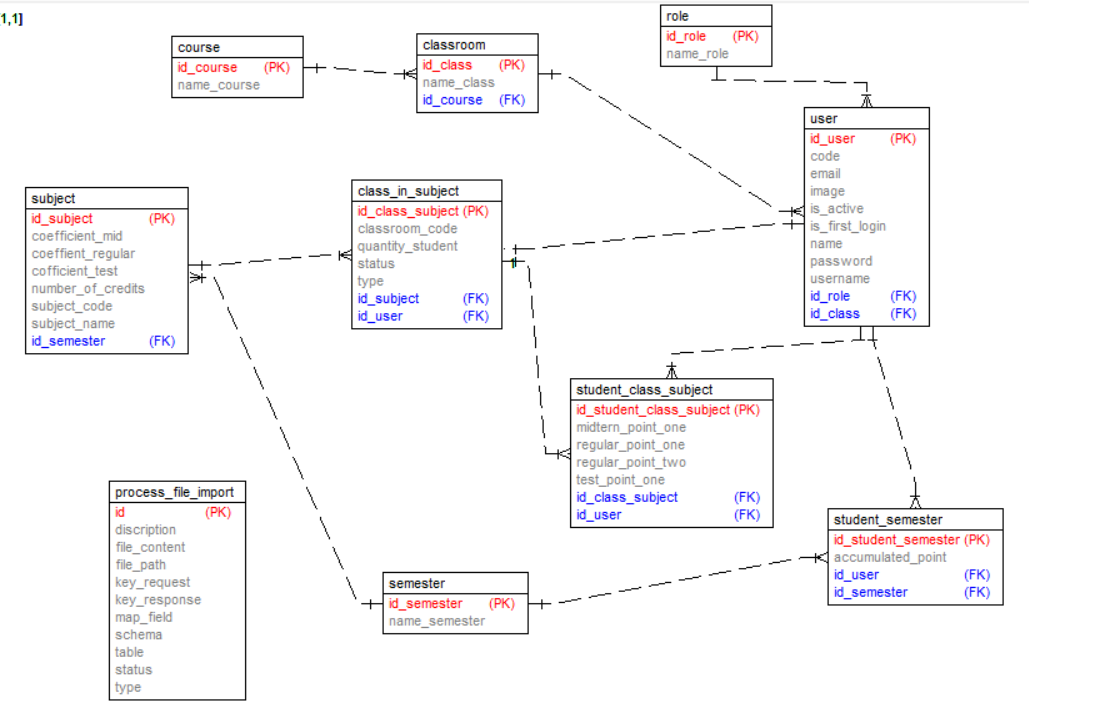
* Giao diện trang web được thiết kế dễ sử dụng và có tính thẩm mỹ cao.
* Được viết trên nền tảng web.
* Hiệu năng: tốc độ, khả năng và độ tin cậy của hệ thống.
* Website bảo mật phân quyền để thực hiện các chức năng của hệ thống.

## Biểu đồ use case tổng quát



Hình 3.1. Biểu đồ use case tổng quát

## . Mô hình thực thể liên kết

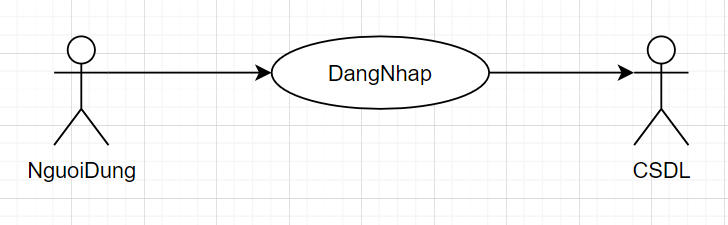


Hình 3.2. Mô hình thực thể liên kết

## Đặc tả use case

### Use case đăng nhập

1. Biểu đồ use case



Hình 3.4. Biểu đồ use case đăng nhập

1. Đặc tả use case

Use case này cho phép người dùng đăng nhập để xác định quyền truy cập vào hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi người dùng bấm vào biểu tượng người dùng sau đó bấm vào nút “Đăng nhập” trên góc phải màn hình.
2. Hệ thống yêu cầu người dùng nhập các thông tin bao gồm: Tên đăng nhập và mật khẩu.
3. Người dùng nhập các thông tin: Tên đăng nhập và mật khẩu và bấm nút “Đăng nhập”.
4. Hệ thống kiểm tra tên đăng nhập và mật khẩu đã nhập trong bảng USER và chuyển hướng sang trang chủ.
5. Use case kết thúc.
   * Luồng rẽ nhánh:
6. Sai tên đăng nhập hoặc mật khẩu: tại bước 3 trong luồng cơ bản nếu người dùng nhập tên đăng nhập hoặc mật khẩu sai, hệ thống hiển thị thông báo lỗi. Người dùng có thể nhập lại hoặc bỏ qua thao tác, khi đó use case kết thúc.
7. Đăng ký: tại bước 3 trong luồng cơ bản, nếu người dùng bấm nút “Đăng ký” use case kết thúc.
8. Quên mật khẩu: Tại bước 3 trong luồng cơ bản, nếu người dùng bấm nút “Quên mật khẩu” use case kết thúc.
9. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

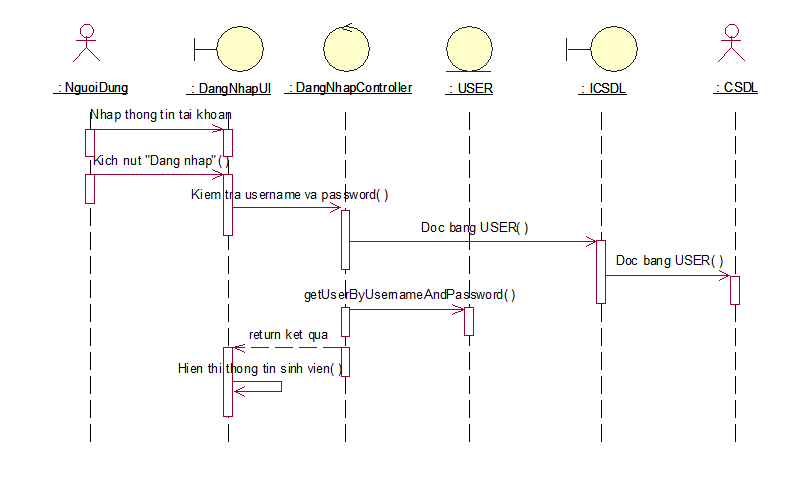
* Hậu điều kiện:

Nếu use case thành công, người dùng sẽ đăng nhập được vào hệ thống. Nếu không trạng thái của hệ thống không thay đổi.

* Điểm mở rộng:

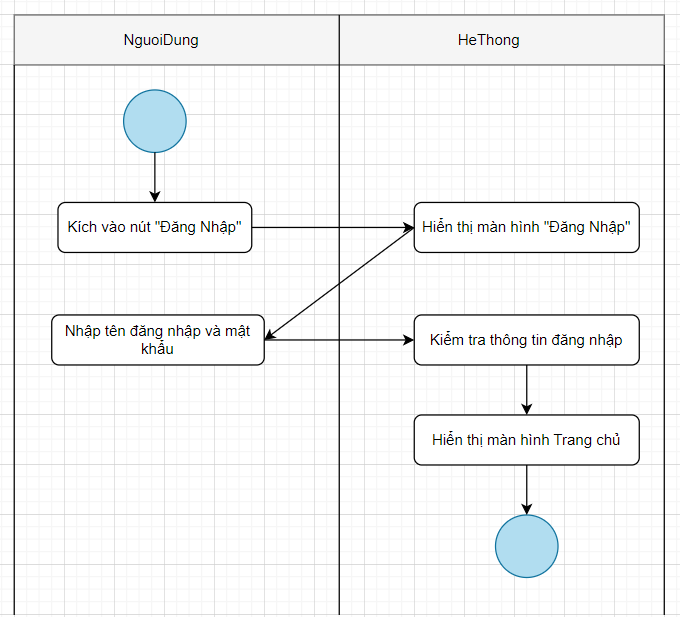
Không có.

1. Biểu đồ trình tự



Hình 3.5. Biểu đồ trình tự use case đăng nhập

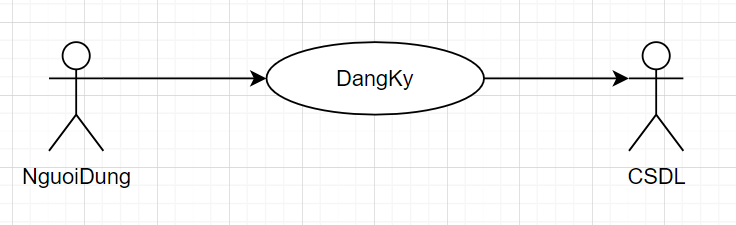
1. Biểu đồ hoạt động



Hình 3.6. Biểu đồ hoạt động use case đăng nhập

### Use case đăng ký

1. Biểu đồ use case



Hình 3.7. Biểu đồ hoạt động use case đăng ký

1. Đặc tả use case

Use case cho phép sinh viên đăng ký tài khoản trên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi sinh viên kích vào nút đăng ký trên màn hình. Hệ thống sẽ hiển thị màn hình đăng ký.
2. Sinh viên nhập thông tin trên form đăng ký gồm Họ tên, Username, Password, Email , Ảnh, Khóa, Lớp rồi kích nút đăng ký. Hệ thống thêm một bản ghi mới vào trong bảng USER trong cơ sở dữ liệu.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Tại bước 2 trong luồng cơ bản, khi nhập thông tin không thoả mãn, hệ thống thông báo lỗi. Use case kết thúc.
5. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

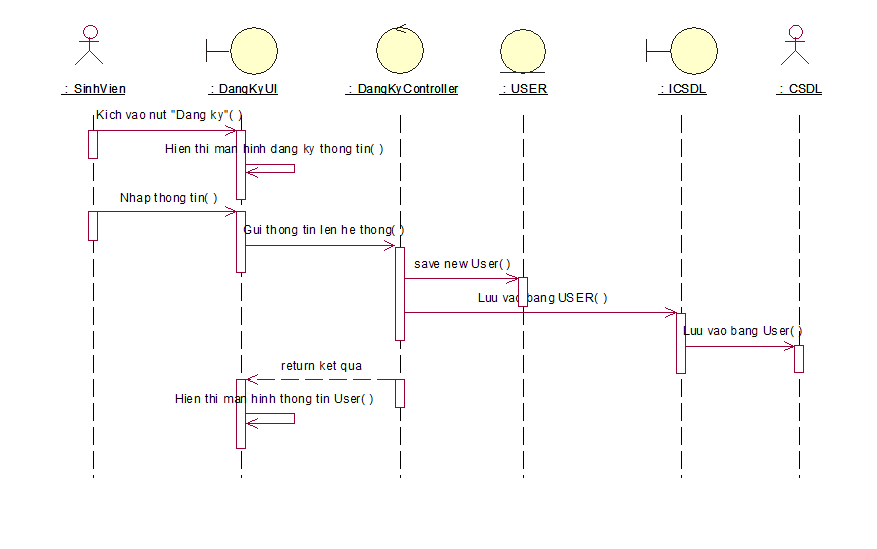
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

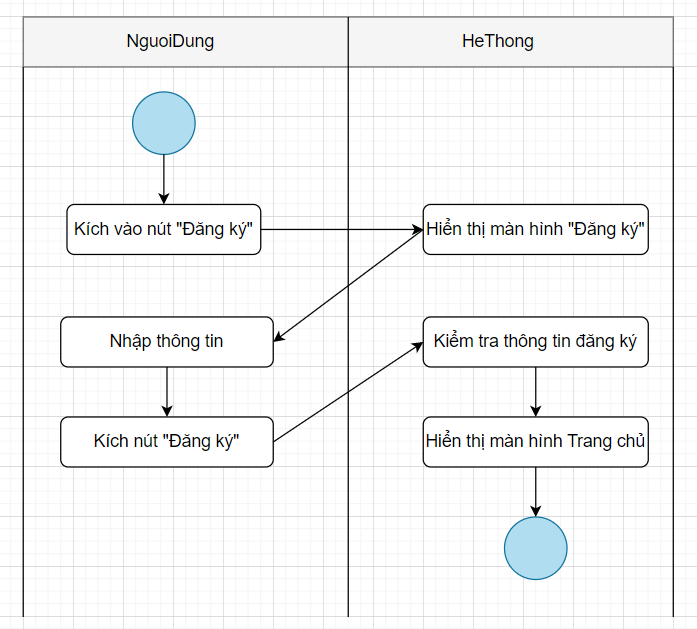
Không có.

1. Biểu đồ trình tự



Hình 3.8. Biểu đồ hoạt động use case đăng ký

1. Biểu đồ hoạt động



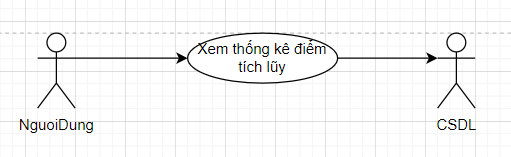
Hình 3.9. Biểu đồ hoạt động use case đăng ký

### Use case Xem thống kê điểm tích lũy

1. Biểu đồ use case
2. Đặc tả use case
3. Biểu đồ trình tự
4. Biểu đồ hoạt động

### Use case Xem thống kê điểm tích lũy

1. Biểu đồ use case



Hình 3.13. Biểu đồ use case Xem điểm tích lũy

1. Đặc tả use case

Use case cho phép người dùng xem điểm tích lũy của sinh viên trên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi sinh viên kích vào nút Xem tích lũy theo kỳ trên màn hình Xem thông tin chi tiết của một sinh viên. Hệ thống sẽ hiển thị màn hình xem tích lũy.
2. Hệ thống lấy thông tin từ bảng STUDENT\_SEMESTER.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

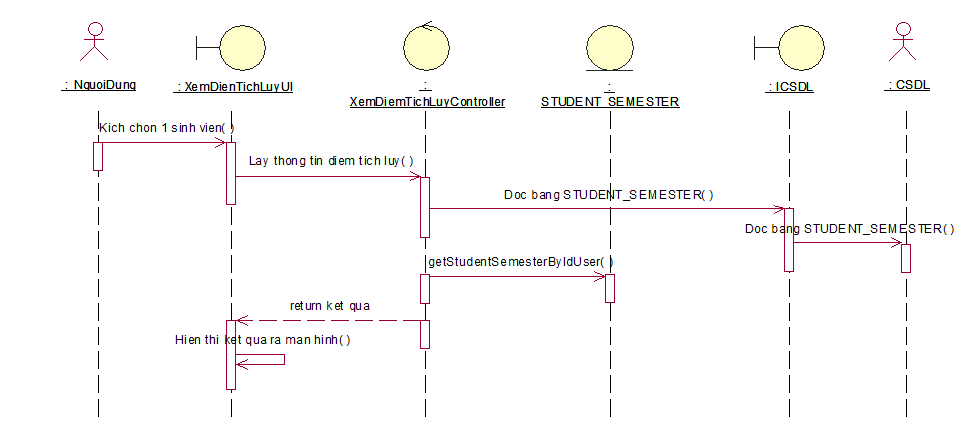
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

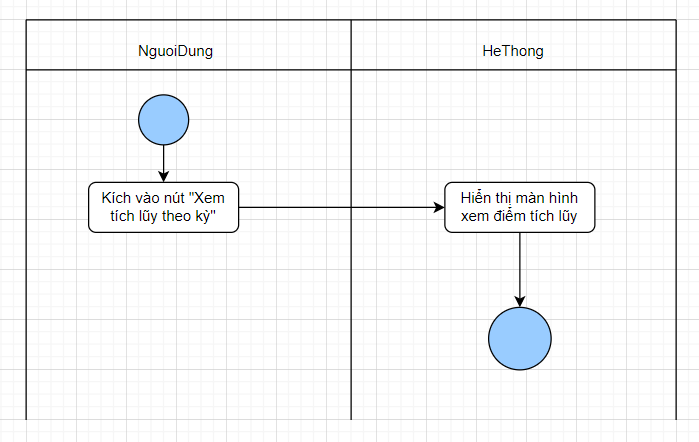
Không có.

1. Biểu đồ trình tự



Hình 3.14. Biểu đồ trình tự use case Xem điểm tích lũy

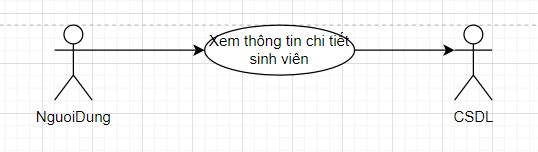
1. Biểu đồ hoạt động



Hình 3.15. Biểu đồ hoạt động use case Xem điểm tích lũy

### Use case Xem thông tin chi tiết sinh viên

1. Biểu đồ use case



Hình 3.16. Biểu đồ use case Xem điểm tích lũy

1. Đặc tả use case

Use case cho phép người dùng xem điểm tích lũy của sinh viên trên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi sinh viên kích một sinh viên bất kỳ. Hệ thống sẽ hiển thị màn hình xem thông tin chi tiết của sinh viên.
2. Hệ thống lấy các thông tin từ bảng SUBJECT, STUDENT\_CLASS\_SUBJECT, CLASS\_IN\_SUBJECT, USER.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

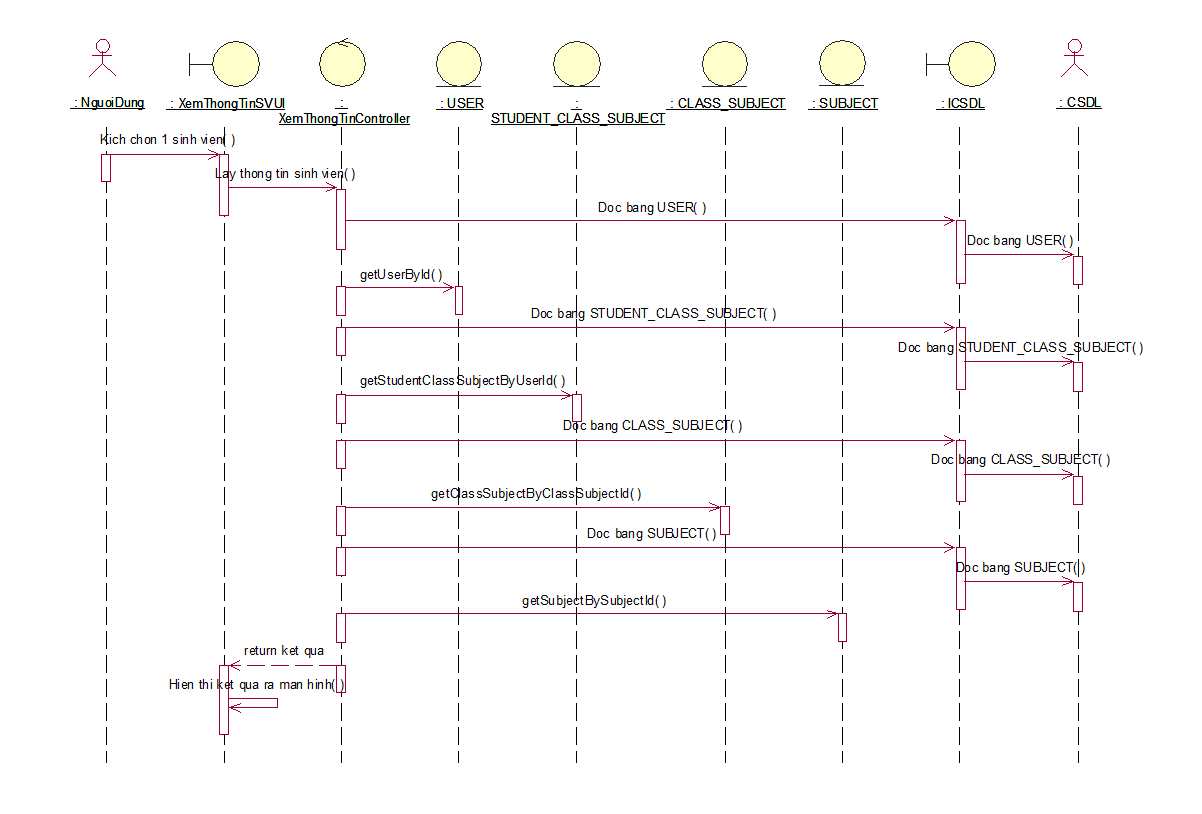
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

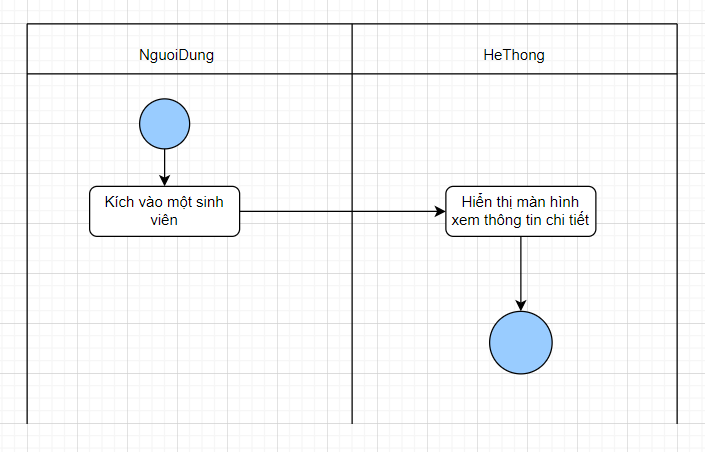
Không có.

1. Biểu đồ trình tự



Hình 3.17. Biểu đồ trình tự use case Xem thông tin chi tiết sinh viên

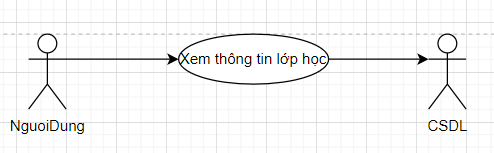
1. Biểu đồ hoạt động



Hình 3.18. Biểu đồ trình tự use case Xem thông tin chi tiết sinh viên

### Use case Xem thông tin lớp học

1. Biểu đồ use case



Hình 3.19. Biểu đồ use case Xem thông tin lớp học

1. Đặc tả use case

Use case cho phép người dùng xem thông tin lớp học của sinh viên trên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi người dùng kích một lớp học bất kỳ của sinh viên trong màn Xem thông tin chi tiết sinh viên. Hệ thống sẽ hiển thị màn hình xem thông tin lớp học.
2. Hệ thống lấy các thông tin từ bảng SUBJECT, STUDENT\_CLASS\_SUBJECT, CLASS\_IN\_SUBJECT, USER.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

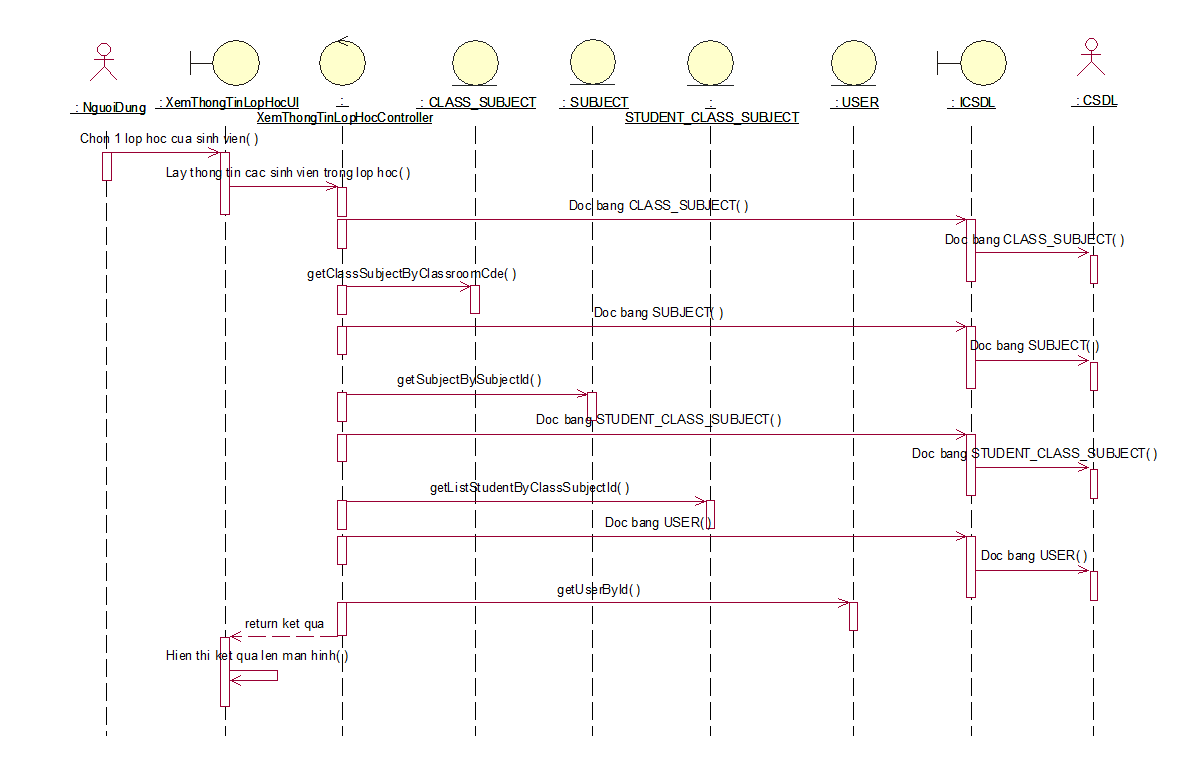
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

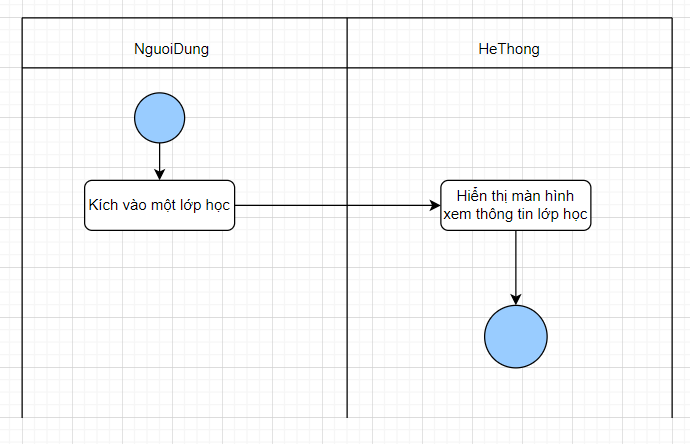
Không có.

1. Biểu đồ trình tự



Hình 3.20. Biểu đồ trình tự use case Xem thông tin lớp học

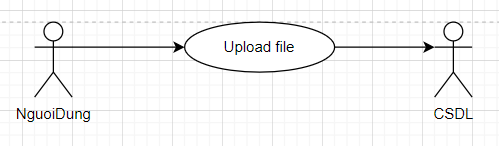
1. Biểu đồ hoạt động



Hình 3.21. Biểu đồ trình tự use case Xem thông tin lớp học

### Use case Upload file

1. Biểu đồ use case



Hình 3.22. Biểu đồ use case Upload file

1. Đặc tả use case

Use case cho phép giảng viên Upload các file excel lên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi giảng viên kích vào nút “Thêm dữ liệu từ file” ở màn nhập điểm.
2. Giảng viên sẽ upload file và Map các trường trong file excel với các trường hiện trên màn hình sau đó ấn “Lưu”
3. Hệ thống sẽ lưu file trong bảng PROCESS\_IMPORT\_FILE.
4. Use case kết thúc.
   * Luồng rẽ nhánh:
5. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

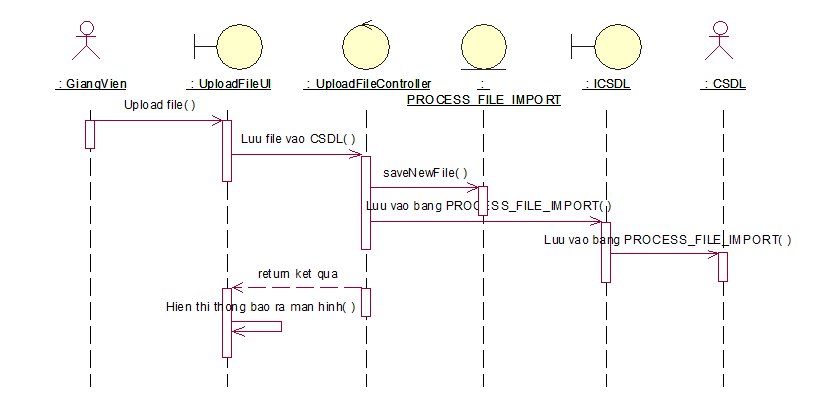
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

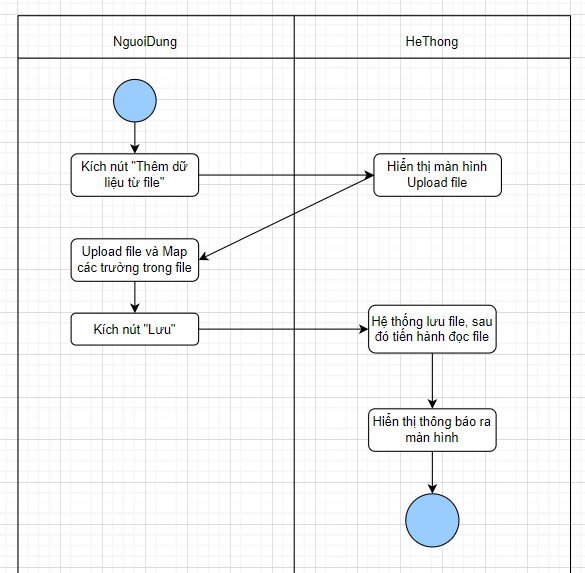
Không có.

1. Biểu đồ trình tự



Hình 3.23. Biểu đồ trình tự use case Upload file

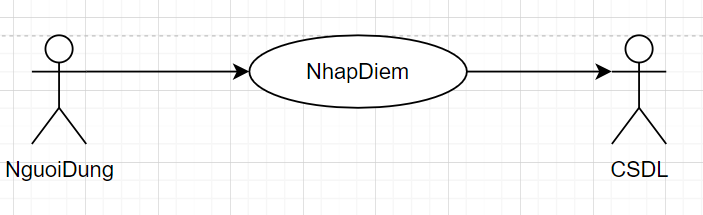
1. Biểu đồ hoạt động



Hình 3.24. Biểu đồ hoạt động use case Upload file

### Use case Nhập điểm

1. Biểu đồ use case



Hình 3.25. Biểu đồ hoạt động use case Nhập điểm

1. Đặc tả use case

Use case cho phép giảng viên Nhập điểm của các sinh viên trong lớp học lên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi giảng viên kích vào một lớp học trong màn “Thông tin môn học”. Hệ thống sẽ lấy ra danh sách sinh viên và điểm trong lớp học đó trong bảng USER, CLASS\_SUBJECT, STUDENT\_CLASS\_SUBJECT.
2. Giảng viên kích vào nút “Sửa” ở bất kỳ sinh viên nào sẽ hiện ra form sửa điểm.
3. Giảng viên nhập điểm và bấn nút “Lưu”. Hệ thống sẽ lưu thông tin vào bảng STUDENT\_CLASS\_SUBJECT.
4. Use case kết thúc.
   * Luồng rẽ nhánh:
5. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

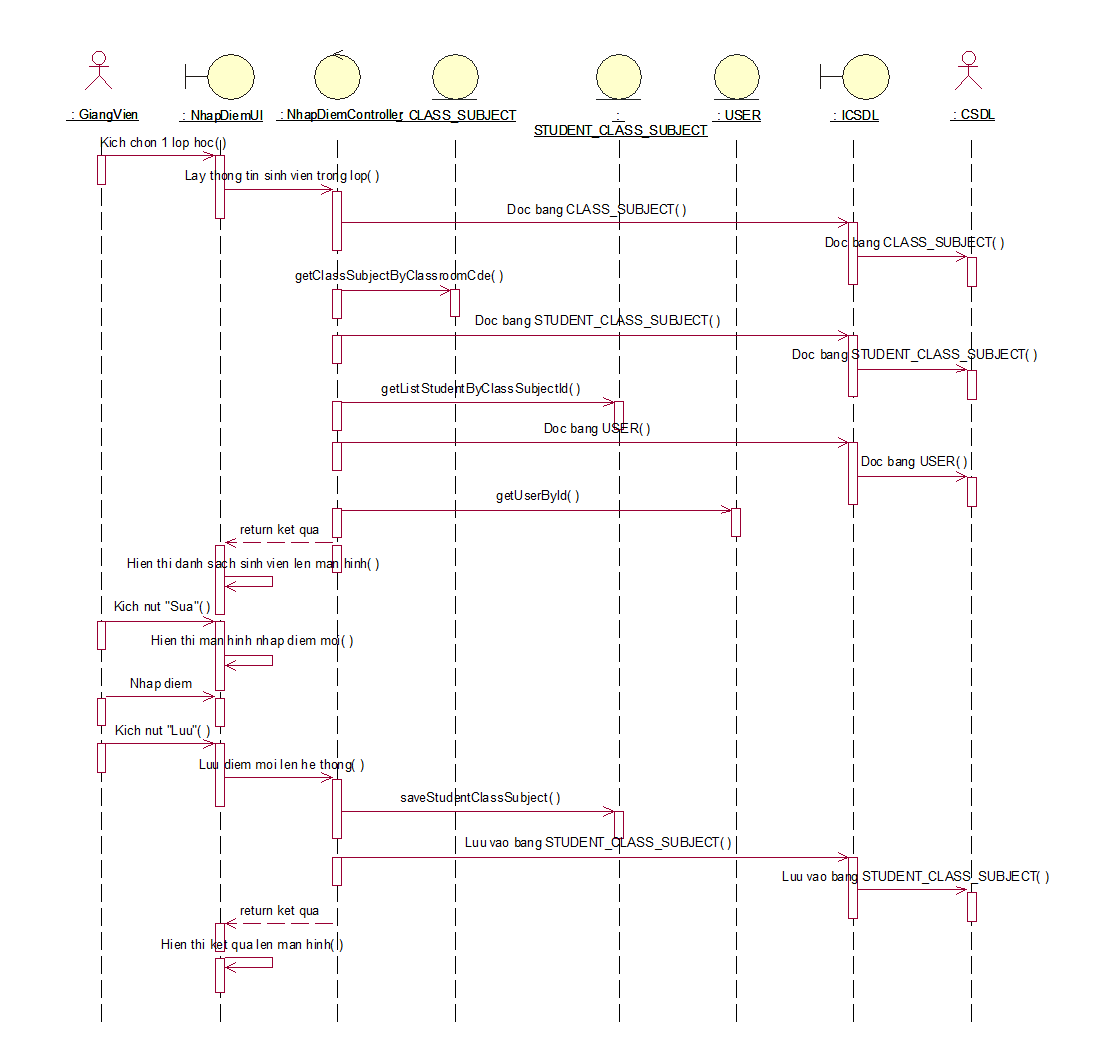
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

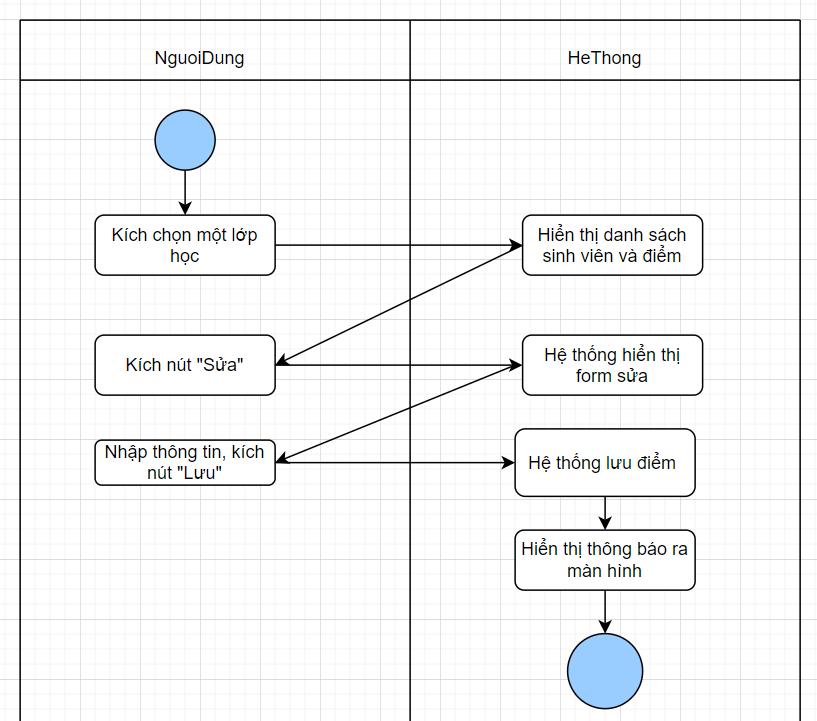
Không có.

1. Biểu đồ trình tự



Hình 3.26. Biểu đồ trình tự use case Nhập điểm

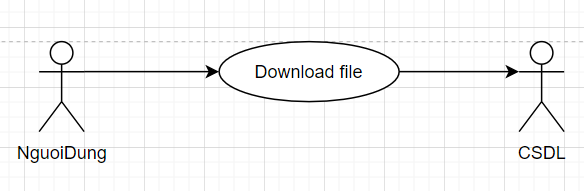
1. Biểu đồ hoạt động



Hình 3.27. Biểu đồ hoạt động use case Nhập điểm

### Use case Download file

1. Biểu đồ use case



Hình 3.28. Biểu đồ use case Download file

1. Đặt tả use case

Use case cho phép giảng viên Download các file excel lên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi giảng viên kích vào “Quản lý trạng thái file”. Hệ thống sẽ lấy danh sách các file đã upload trong bảng PROCESS\_FILE\_IMPORT hiển thị ra màn hình.
2. Giảng viên kích nút “Download” để download file về thiết bị.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

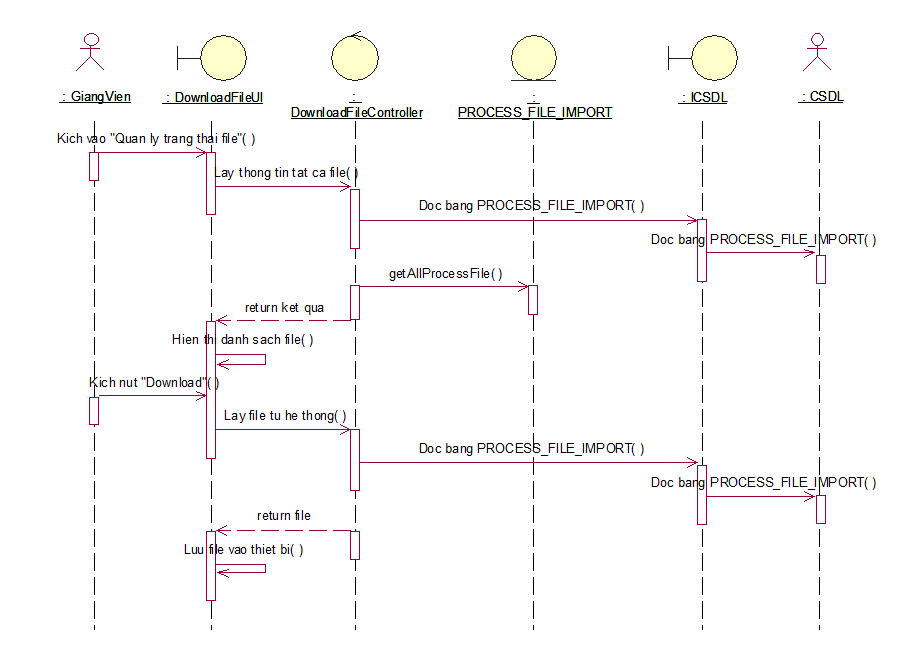
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

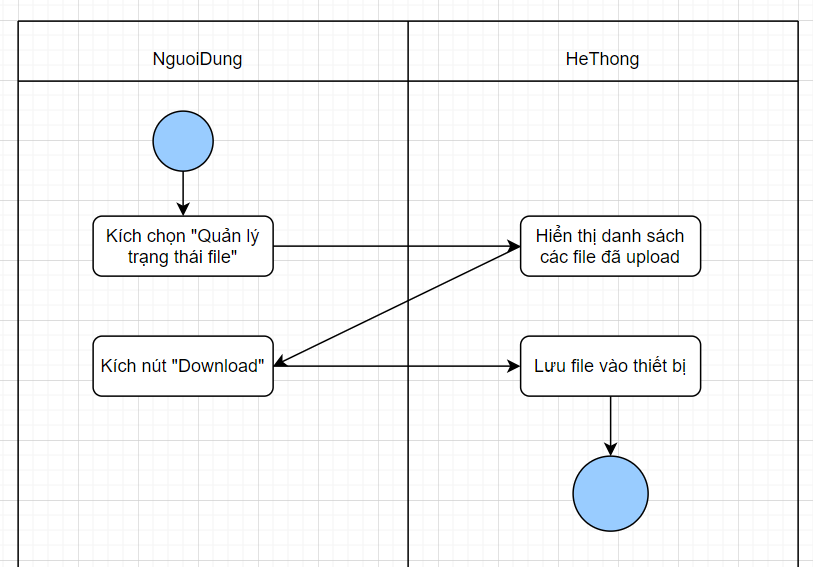
Không có.

1. Biểu đồ trình tự



Hình 3.29. Biểu đồ trình tự use case Download file

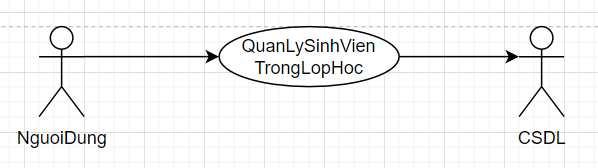
1. Biểu đồ hoạt động



Hình 3.30. Biểu đồ hoạt động use case Download file

### Use case Quản lý sinh viên trong lớp học

1. Biểu đồ use case



Hình 3.31. Biểu đồ hoạt động use case Download file

1. Đặt tả use case

Use case cho phép admin thêm sinh viên vào lớp học

* Luồng sự kiện:
  + Luồng cơ bản:

1. Use case này bắt đầu khi Admin kích vào nút “Thêm sinh viên. Hệ thống sẽ hiển thị form nhập thông tin.
2. Admin sẽ nhập mã sinh viên của sinh viên. Hệ thống sẽ truy cập vào bảng USER lấy ra các thông tin khác với mã sinh viên. Hiển thị ra màn hình.
3. Admin kích nút “Thêm”, sinh viên đó sẽ được thêm vào lớp học và lưu trong bảng STUDENT\_CLASS\_SUBJECT.
4. Use case kết thúc.
   * Luồng rẽ nhánh:
5. Nếu admin nhập mã sinh viên không có trong bảng USER thì hệ thống sẽ hiện thông báo.
6. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

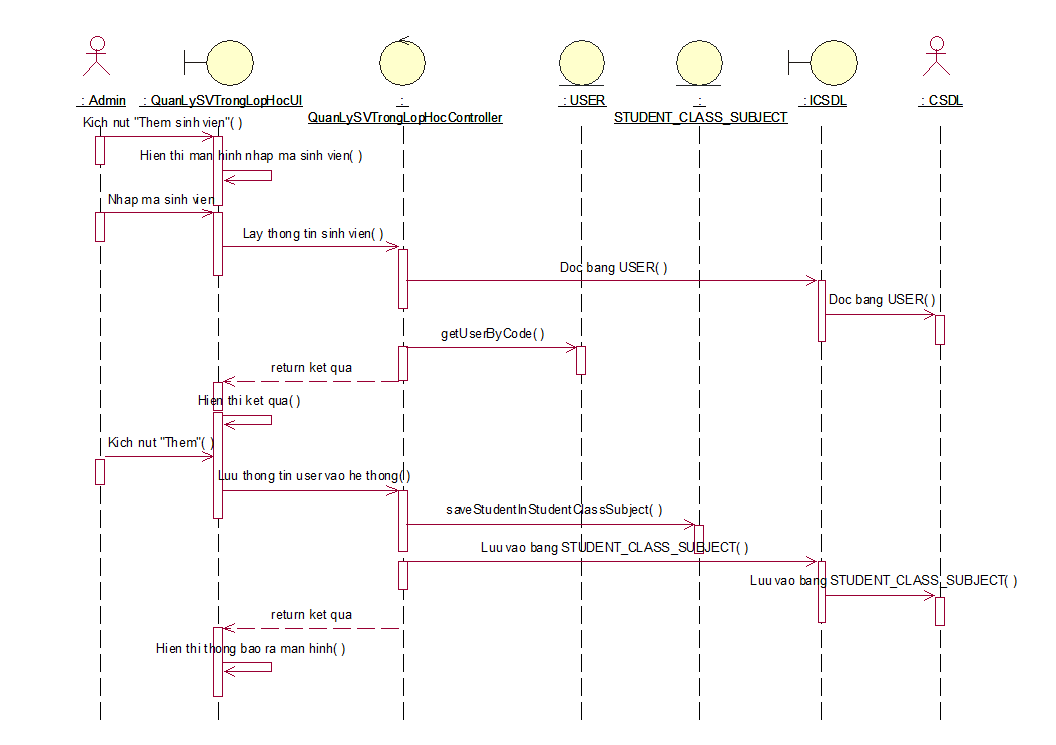
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

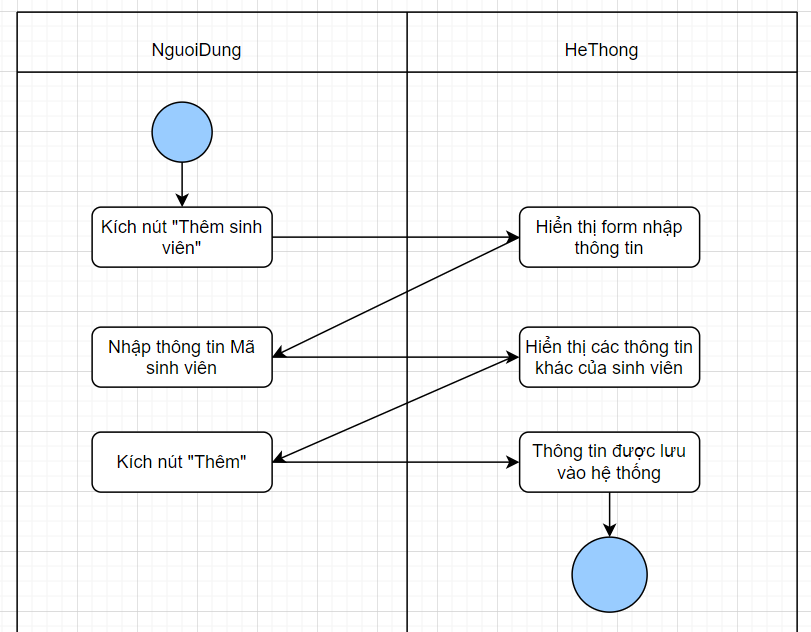
Không có.

1. Biểu đồ trình tự



Hình 3.32. Biểu đồ hoạt động use case Quản lý sinh viên trong lớp học

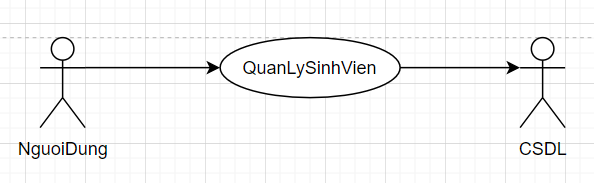
1. Biểu đồ hoạt động



Hình 3.33. Biểu đồ hoạt động use case Quản lý sinh viên trong lớp học

### Use case Quản lý sinh viên

1. Biểu đồ use case

 Hình 3.34. Biểu đồ use case Quản lý sinh viên

1. Đặc tả use case

Use case cho phép admin Thêm, Sửa thông tin sinh viên trên hệ thống.

* Luồng sự kiện:
  + Luồng cơ bản:
    - Thêm sinh viên

1. Use case này bắt đầu khi Admin kích vào “Quản lý sinh viên”. Hệ thống sẽ hiển thị ra màn hình Quản lý sinh viên.
2. Kích vào nút “Thêm sinh viên” trên màn hình. Hệ thống hiển thị ra form nhập thông tin.
3. Hệ thống sẽ lưu thông tin vào bảng USER và hiển thị thông báo lên màn hình.
4. Use case kết thúc.

* Sửa sinh viên

1. Admin kích vào nút “Sửa” trên màn hình. Hệ thống hiển thị form sửa.
2. Admin nhập thông tin và ấn nút “Lưu”. Hệ thống sẽ cập nhật thông tin mới vào bảng USER.
3. Use case kết thúc.
   * Luồng rẽ nhánh:
4. Nếu admin nhập username đã tồn tại trong bảng USER thì hệ thống sẽ thông báo lỗi.
5. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi và use case kết thúc.

* Các yêu cầu đặc biệt:

Không có.

* Tiền điều kiện:

Không có.

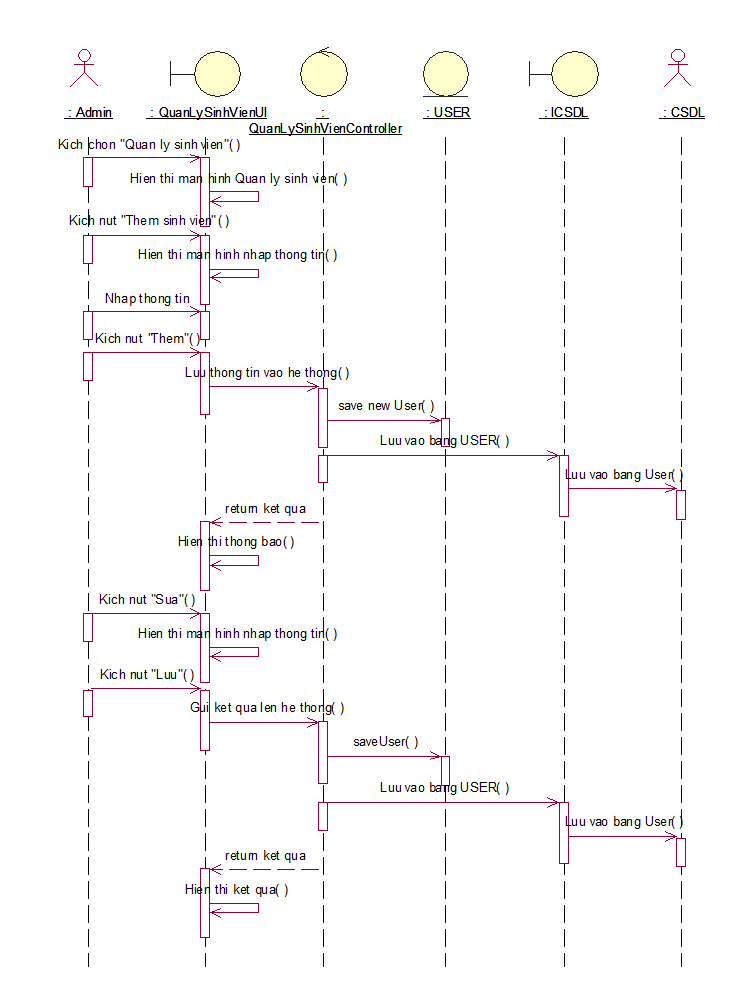
* Hậu điều kiện:

Không có.

* Điểm mở rộng:

Không có.

1. Biểu đồ trình tự



*Hình 3.34. Biểu đồ use case Quản lý sinh viên*

1. Biểu đồ hoạt động