**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**



**TÊN ĐỀ TÀI MÔN KIẾN TRÚC VÀ THIẾT KẾ PHẦN MỀM**

***Nhóm 34 - Sinh viên thực hiện***

1. Nguyễn Thành Doanh - 20054911
2. Dương Tuấn Kiệt - 20001545
3. Vũ Thế Kiệt - 20001565

# MỤC LỤC

[MỤC LỤC 1](#_Toc14693502)

[DANH MỤC CÁC HÌNH VẼ 1](#_Toc14693503)

[DANH MỤC CÁC BẢNG BIỂU 1](#_Toc14693504)

[CHƯƠNG 1 : GIỚI THIỆU 1](#_Toc14693505)

[1.1 Tổng quan 1](#_Toc14693506)

[1.2 Mục tiêu đề tài 1](#_Toc14693507)

[1.3 Phạm vi đề tài 1](#_Toc14693508)

[1.4 Mô tả yêu cầu chức năng 1](#_Toc14693509)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 1](#_Toc14693510)

[2.1 Microservice Architecture 1](#_Toc14693511)

[2.2 Spring boot 1](#_Toc14693512)

[2.3 Docker 1](#_Toc14693513)

[2.4 MongoDB 1](#_Toc14693513)

[2.5 HeiSQL 1](#_Toc14693513)

[CHƯƠNG 3 : PHÂN TÍCH VÀ THIẾT KẾ 1](#_Toc14693514)

[3.1 Phân tích yêu cầu bằng UML 1](#_Toc14693515)

[3.1.1 Usecase tổng quát 1](#_Toc14693516)

[3.1.2 Danh sách tác nhân và mô tả 1](#_Toc14693517)

[3.1.3 Danh sách các tình huống hoạt động (Use cases) 1](#_Toc14693518)

[3.1.4 Tình huống hoạt động 1](#_Toc14693519)

[3.2 Class diagram 1](#_Toc14693520)

[3.3 Deployment diagram 1](#_Toc14693521)

[CHƯƠNG 4 : HIỆN THỰC 1](#_Toc14693522)

[4.1 Cấu hình phần cứng, phần mềm 1](#_Toc14693523)

[4.2 Giao diện của hệ thống 1](#_Toc14693524)

[CHƯƠNG 5 : KẾT LUẬN 1](#_Toc14693525)

[5.1 Kết quả đạt được 1](#_Toc14693526)

[5.2 Hạn chế của đồ án 1](#_Toc14693527)

[5.3 Hướng phát triển 1](#_Toc14693528)

[TÀI LIỆU THAM KHẢO 1](#_Toc14693529)

# DANH MỤC CÁC HÌNH VẼ

[Hình 2‑1 Node.js development 1](#_Toc14693413)

# DANH MỤC CÁC BẢNG BIỂU

[Bảng 3‑1 Danh sách tác nhân và mô tả 1](#_Toc14682192)

[Bảng 3‑2 Danh sách các tình huống trong hệ thống 1](#_Toc14682193)

**LỜI MỞ ĐẦU**

Trong thế giới hiện nay công nghệ thông tin ngày càng phát triển vượt bậc và ngày càng đạt được thành tựu to lớn trong việc phát triển kinh tế. Trên hầu hết tất cả lĩnh vực thì đều có mặt ngành công nghệ thông tin trong đó, nó đã trở thành một phần thiết yếu trong cuộc sống. Nhờ có công nghệ thông tin mà kinh tế có thể phát triển và đời sống con người được nâng cao, xã hội ngày càng tiên tiến. Việc tận dụng công nghệ thông tin vào mọi ngành nghề đã không còn là một việc quá xa lạ.

Đăng ký học phần là một trong những nhiệm vụ quan trọng của bất kỳ tổ chức giáo dục nào. Việc thu thập, lưu trữ và xử lý thông tin về sinh viên đòi hỏi sự chính xác, hiệu quả và dễ dàng truy cập để hỗ trợ quá trình quản lý và đưa ra các quyết định phù hợp.

Với sự phát triển không ngừng của công nghệ thông tin, việc sử dụng các công cụ và kỹ thuật công nghệ mới trong việc đăng ký học phần đã trở nên cần thiết. Trong đó, sự kết hợp giữa ngôn ngữ lập trình Java, và cơ sở dữ liệu Mariadb Workbench là một phương pháp phổ biến và mạnh mẽ để tạo ra một hệ thống đăng ký học phần linh hoạt, mạnh mẽ và dễ dàng mở rộng.

Với đề tài này giúp chúng ta củng cố lại những kiến thức về công nghệ. Đồ án thực hiện dựa trên những kiến thức đã học và tìm kiếm trên Internet. Do kiến thức và trình độ còn non kém nên chúng em chưa hoàn thành đầy đủ các công tác quản lý. Trong quá trình thực hiện nếu có sai sót mong thầy bỏ qua.

Chúng em xin cam đoan đồ án: “Đăng ký học phần” là công trình nghiên cứu của bản thân. Những phần sử dụng tài liệu tham khảo trong luận văn đã được nêu rõ trong phần tài liệu tham khảo. Các số liệu, kết quả trình bày trong luận văn là hoàn toàn trung thực, nếu sai chúng em xin chịu hoàn toàn trách nhiệm và chịu mọi kỷ luật của bộ môn và nhà trường đề ra.

# : GIỚI THIỆU

## Tổng quan

Đăng ký học phần là một phần quan trọng trong hệ thống giáo dục, đặc biệt là trong các cơ sở giáo dục lớn với số lượng sinh viên đông đảo. Để quản lý và theo dõi thông tin của sinh viên một cách hiệu quả, việc xây dựng các hệ thống đăng ký học phần đã trở thành một xu hướng không thể phủ nhận.

Chương này giới thiệu tổng quan về đề tài đăng ký học phần, nhằm cung cấp một cái nhìn tổng quan về nội dung và mục tiêu của nghiên cứu. Chúng ta sẽ tìm hiểu về tình hình hiện tại của đăng ký học phần, các vấn đề và thách thức mà các tổ chức giáo dục đang phải đối mặt, cũng như ý nghĩa và lợi ích của việc xây dựng một hệ thống đăng ký học phần hiệu quả.

Chương này cũng sẽ trình bày về phạm vi và phương pháp nghiên cứu được sử dụng để đạt được mục tiêu của đề tài. Bằng việc nắm bắt các khía cạnh quan trọng của đăng ký học phần và áp dụng các phương pháp nghiên cứu phù hợp, chúng ta hy vọng sẽ đưa ra các giải pháp và gợi ý để nâng cao hiệu quả việc đăng ký học phần trong các tổ chức giáo dục.

Mục tiêu của chương này là cung cấp một cái nhìn tổng quan về đề tài đăng ký học phần và tạo nền tảng cho các phần tiếp theo của báo cáo nghiên cứu. Bằng cách hiểu rõ về tình hình hiện tại, mục tiêu, phạm vi và phương pháp nghiên cứu, chúng ta sẽ có cơ sở để tiếp cận và nghiên cứu chi tiết hơn về các khía cạnh cụ thể của đăng ký học phần và xây dựng một hệ thống đăng ký học phần hiệu quả.

## Mục tiêu đề tài:

Tối ưu hóa quản lí thông tin đăng ký học phần: Phát triển và triển khai các phương pháp và công nghệ để quản lí thông tin đăng ký học phần một cách hiệu quả. Xây dựng một hệ thống cơ sở dữ liệu đăng ký học phần toàn diện và tin cậy, giúp cải thiện quá trình quản lí đăng ký, thời khóa biểu và các thông tin khác liên quan đến học phần.

Tăng cường tương tác giữa sinh viên và tổ chức giáo dục: Xây dựng các công cụ và giao diện tương tác để tăng cường tương tác giữa sinh viên và tổ chức giáo dục. Cung cấp các kênh liên lạc hiệu quả, như ứng dụng di động và hệ thống thông báo, giúp sinh viên và nhân viên có thể trao đổi thông tin và nhận thông báo quan trọng một cách thuận tiện và nhanh chóng.

Theo dõi tiến trình đăng ký học phần và kết quả: Phát triển các công cụ và cơ chế để theo dõi tiến trình đăng ký học phần và kết quả của sinh viên. Cung cấp cho sinh viên và tổ chức giáo dục một cái nhìn toàn diện về tiến trình đăng ký và kết quả học tập, nhằm đảm bảo sự theo dõi và hỗ trợ tốt nhất cho sinh viên trong quá trình học tập.

Tối ưu hóa quy trình đăng ký học phần: Nghiên cứu và cải tiến các quy trình đăng ký học phần hiện có để tối ưu hóa hiệu suất và hiệu quả. Tạo ra một quy trình đăng ký học phần linh hoạt, nhanh chóng và tiết kiệm thời gian cho cả sinh viên và tổ chức giáo dục.

Tạo ra một trải nghiệm học tập tích hợp và thuận tiện, giúp sinh viên quản lý thông tin và thời gian học tập một cách hiệu quả, bao gồm cung cấp thông tin về số tín chỉ đã đạt, số môn đã học, điểm môn học và điểm trung bình tích lũy.

Cho phép sinh viên đăng ký xét tốt nghiệp sau khi đạt đủ số tín chỉ theo từng ngành. Nếu sinh viên đáp ứng đầy đủ các tiêu chuẩn, họ sẽ được cấp bằng tốt nghiệp và thông tin về bằng cấp sẽ được công khai trên trang web của nhà trường. Thông tin của cựu sinh viên sẽ được lưu giữ để theo dõi quá trình làm việc và tham gia các hoạt động khác, với sự đồng ý của sinh viên.

Tổng quan, mục tiêu của đề tài là xây dựng một hệ thống đăng ký học phần hiệu quả, sử dụng mô hình client-server, nhằm cải thiện quá trình đăng ký, quản lí thông tin và tương tác giữa sinh viên và tổ chức giáo dục. Điều này giúp tạo điều kiện thuận lợi cho sinh viên trong quá trình học tập và đồng thời giúp các tổ chức giáo dục nâng cao hiệu suất và hiệu quả trong quản lí học phần.

## Phạm vi đề tài:

* Phần mềm sẽ tập trung vào các khía cạnh sau:

+ Đăng ký học phần

+ Lịch học

+ Thông tin học tập cá nhân

+ Môn học

+ Đăng ký xét tốt nghiệp

## Mô tả yêu cầu chức năng

* Đăng ký học phần:

+ Sinh viên có thể đăng ký học phần thông qua giao diện người dùng.

+ Hệ thống tự động kiểm tra các điều kiện tiên quyết trước khi cho phép sinh viên đăng ký.

+ Hạn chế số lượng sinh viên đăng ký vào mỗi lớp học tín chỉ để đảm bảo không quá tải.

* Thời khóa biểu:

+ Sinh viên và giảng viên có thể xem và quản lý thời khóa biểu của mình.

+ Hệ thống cập nhật thời khóa biểu tự động khi có thay đổi trong đăng ký học phần.

+ Tra cứu thông tin học tập:

+ Sinh viên có thể tra cứu thông tin về các môn học đã đăng ký, điểm số, và tình trạng học tập.

+ Giảng viên có thể tra cứu thông tin về các lớp học mình giảng dạy và điểm số của sinh viên.

* Xác nhận và đăng ký:

+ Sinh viên phải xác nhận đăng ký học phần trước khi hệ thống chấp nhận đăng ký của họ.

+ Sinh viên có thể quản lý và chỉnh sửa đăng ký của mình trước thời hạn đăng ký.

* Thông tin sinh viên:

+ Hệ thống lưu trữ và thông tin cá nhân của sinh viên, bao gồm hồ sơ học tập, điểm số, và tiến trình học tập.

* Tính năng quản trị:

+ Quản trị viên có thể quản lý danh sách môn học, lớp học, thông tin sinh viên, và các thiết lập hệ thống khác.

+ Hệ thống cung cấp giao diện quản trị dễ sử dụng cho việc quản lý hệ thống và xử lý các vấn đề kỹ thuật.

# : CƠ SỞ LÝ THUYẾT

## Client - Server architecture:

* Khái niệm: Kiến trúc Client – Server là mô hình mà trong đó các máy tính như máy chủ cung cấp dịch vụ mạng cho các máy tính khác như máy khách để thực hiện các tác vụ dựa trên người dùng.
* Client: là một chương trình chạy trên máy cục bộ yêu cầu dịch vụ từ máy chủ. Chương trình khách là một chương trình hữu hạn có nghĩa là dịch vụ được người dùng bắt đầu và chấm dứt khi dịch vụ hoàn thành.
* Server: là một chương trình chạy trên máy từ xa cung cấp dịch vụ cho khách hàng. Khi máy khách yêu cầu một dịch vụ, máy chủ sẽ mở cửa cho các yêu cầu đến nhưng không bao giờ khởi tạo dịch vụ.
* Ưu điểm:

+ Tập trung: Có thể sao lưu tập trung trong mạng máy khách-máy chủ, tức là tất cả dữ liệu được lưu trữ trong máy chủ.

+ Bảo mật: Các mạng này an toàn hơn vì tất cả tài nguyên dùng chung đều được quản lý tập trung.

+ Hiệu suất: Việc sử dụng máy chủ chuyên dụng giúp tăng tốc độ chia sẻ tài nguyên. Điều này làm tăng hiệu suất của hệ thống tổng thể.

+ Khả năng mở rộng: Chúng tôi có thể tăng số lượng máy khách và máy chủ một cách riêng biệt, tức là có thể thêm phần tử mới hoặc chúng tôi có thể thêm một nút mới trong mạng bất kỳ lúc nào.

* Nhược điểm:

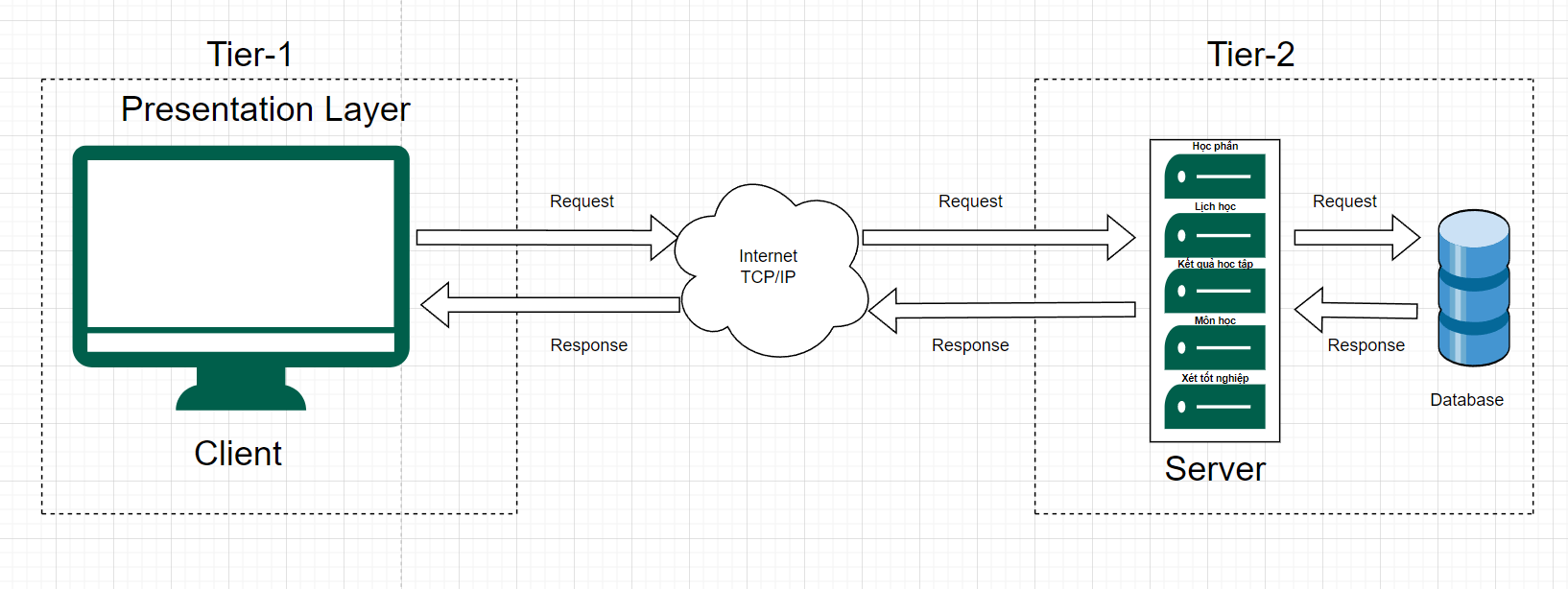
+ Tắc nghẽn là một vấn đề lớn trong mạng Client/Server. Khi một số lượng lớn khách hàng gửi yêu cầu đến cùng một máy chủ có thể gây ra vấn đề ùn tắc.

+ Nó không có sự mạnh mẽ của mạng, tức là khi máy chủ ngừng hoạt động thì các yêu cầu của khách hàng không thể được đáp ứng.

+ Mạng máy khách/máy chủ có tính quyết định rất lớn. Đôi khi, phần cứng máy tính thông thường không phục vụ được một số lượng khách hàng nhất định. Trong những tình huống như vậy, phía máy chủ cần có phần cứng cụ thể để hoàn thành công việc.

+ Đôi khi tài nguyên tồn tại trên máy chủ nhưng có thể không tồn tại trên máy khách. Ví dụ: Nếu ứng dụng là web thì chúng ta không thể in trực tiếp trên máy in mà không thoát khỏi cửa sổ xem bản in trên web.

* Sơ đồ kiến trúc client – server (2 tier):



## Spring boot:

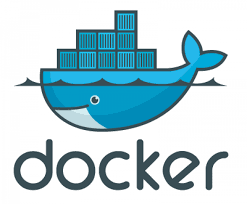
* Khái niệm: Spring Boot là một dự án phát triển bởi JAV (ngôn ngữ java) trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên chúng ta đơn giản hóa quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển business cho ứng dụng.



Hình ‑ Spring boot

## Docker:

* Khái niệm: Docker là một nền tảng cho developers và system admin để develop, deploy và run application với container. Nó cho phép tạo các môi trường độc lập và tách biệt để khởi chạy và phát triển ứng dụng và môi trường này được gọi là container. Khi cần deploy lên bất kỳ server nào chỉ cần run container của Docker thì application của bạn sẽ được khởi chạy ngay lập tức.



## MongoDB:

* Khái niêm: MongoDB là một cơ sở dữ liệu document-oriented được sử dụng để lưu trữ dữ liệu có volume lớn. Thay vì sử dụng các tables và rows như trong cơ sở dữ liệu relational truyền thống, MongoDB sử dụng collections và documents.



## HeiSQL:

* Khái niệm: HeidiSQL là một công cụ quản lý database tất cả trong một. Có thể dùng HeidiSQL để kết nối từ xa tới database được tạo trên Hostinger.



# : PHÂN TÍCH VÀ THIẾT KẾ

## Phân tích yêu cầu bằng UML

### Usecase tổng quát

******

### Danh sách tác nhân và mô tả

Danh sách tác nhân và mô tả xem ở Bảng 3-1.

|  |  |
| --- | --- |
| Tác nhân | Mô tả tác nhân |
| User | User được xác định thông qua tài khoản đăng nhập, bao gồm mã số sinh viên, mật khẩu, user có thể đăng ký học phần, hủy học phần mình đã đăng ký, xem danh sách môn mình đã đăng ký, xem lịch học, xem điểm, đóng học phí bằng cách thanh toán trực tiếp trên trang web & đăng ký xét tốt nghiệp. |

Bảng ‑ Danh sách tác nhân và mô tả

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

|  |  |
| --- | --- |
| ID | Tên Use case |
| UC001 | Đăng nhập |
| UC002 | Đăng ký học phần mới |
| UC003 | Hủy học phần vừa đăng ký |
| UC004 | Xem danh sách môn học đã đăng ký học một kỳ |
| UC005 | Xem lịch học |
| UC006 | Xem điểm |
| UC007 | Đóng học phí Onlline |
| UC008 | Đăng ký xét tốt nghiệp |

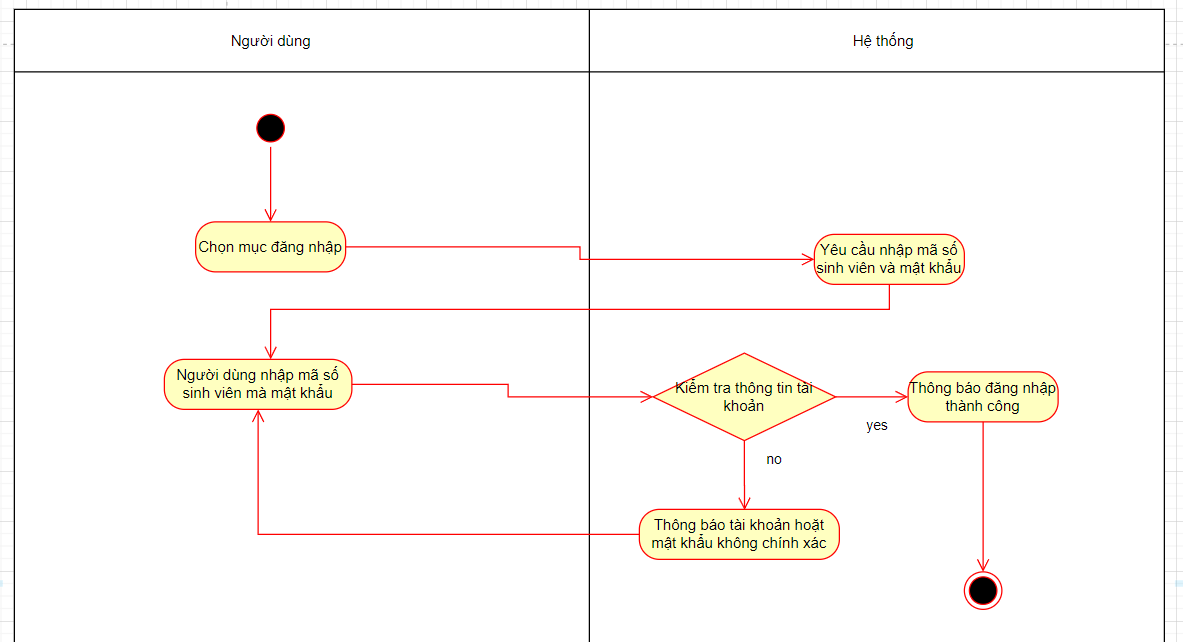
Bảng ‑ Danh sách các tình huống trong hệ thống

### Tình huống hoạt động

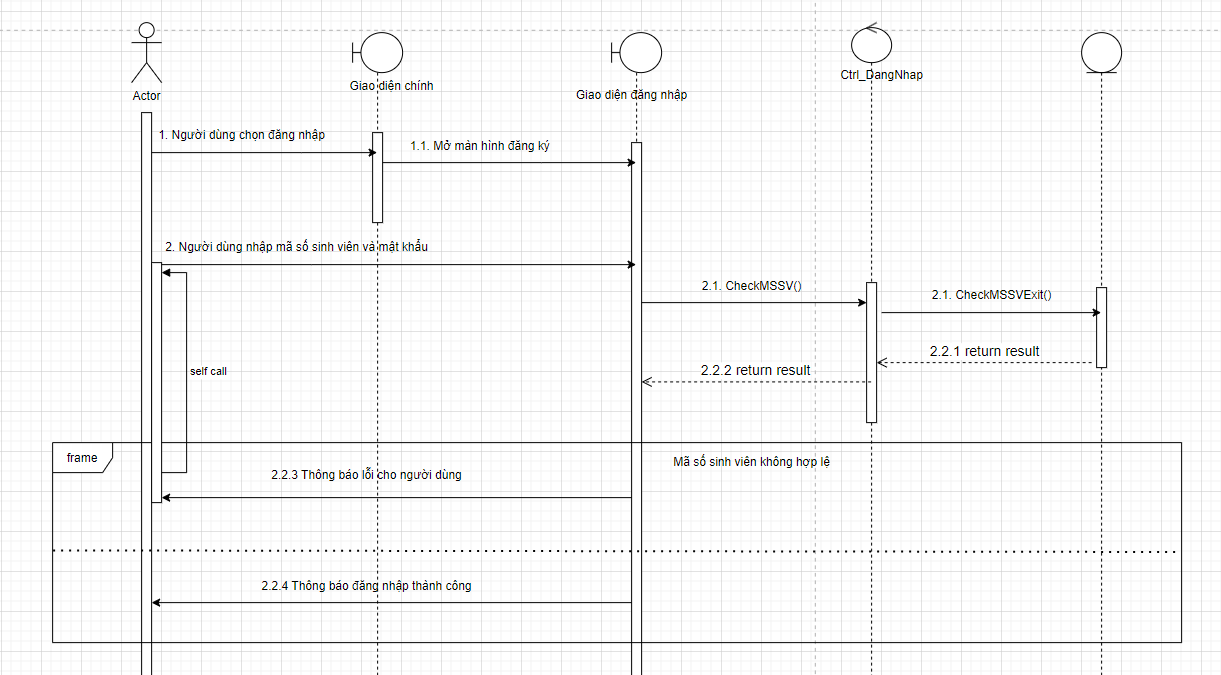
Đặc tả use case.

|  |  |  |
| --- | --- | --- |
| **UC001\_Đăng Nhập** | | |
| **Mục đích** | Cho phép người dùng đăng nhập tài khoản | |
| **Mô tả** | Usecase này mô tả cách người dùng đăng nhập tài khoản | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng sở hữu tài khoản | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn mục đăng nhập | | 2. Hệ thống yêu cầu người dùng nhập mã số sinh viên và mật khẩu |
| 3. Người dùng nhập mã số sinh viên và mật khẩu bấm đăng nhập | | 4.Hệ thống kiểm tra thông tin tài khoản |
|  | | 5.Hệ thống thông báo đăng nhập thành công |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |
|  | | 4.1 Nếu mã số sinh viên hoặc mật khẩu không hợp lệ thì hệ thống báo lỗi. |
| 4.2 Người dùng quay về bước 3 | |  |

Activity diagram.

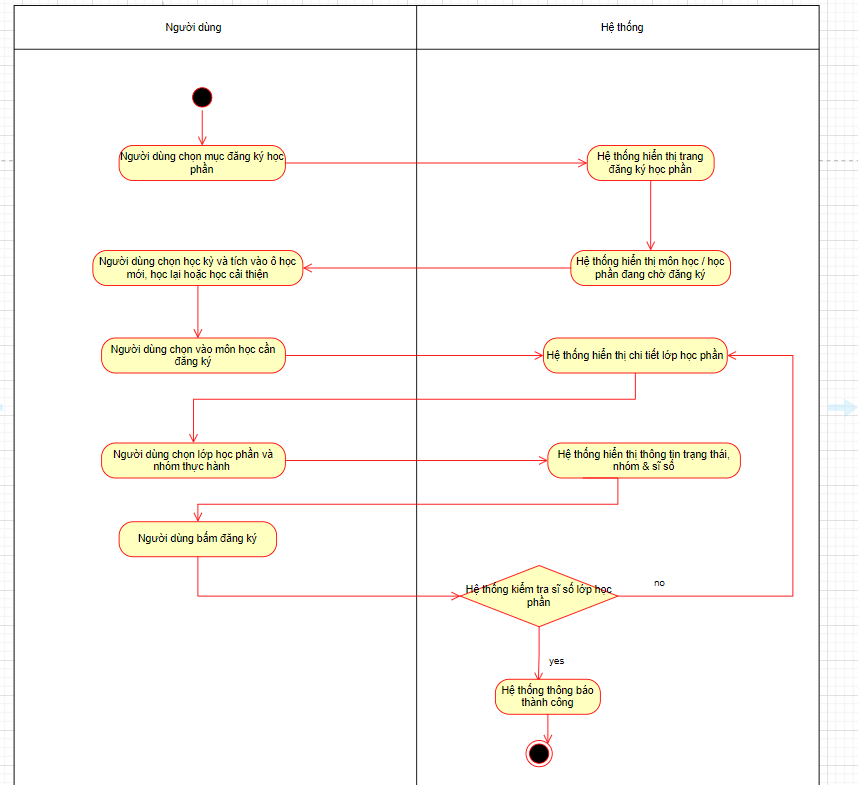


Sequence diagram cho từng tình huống.

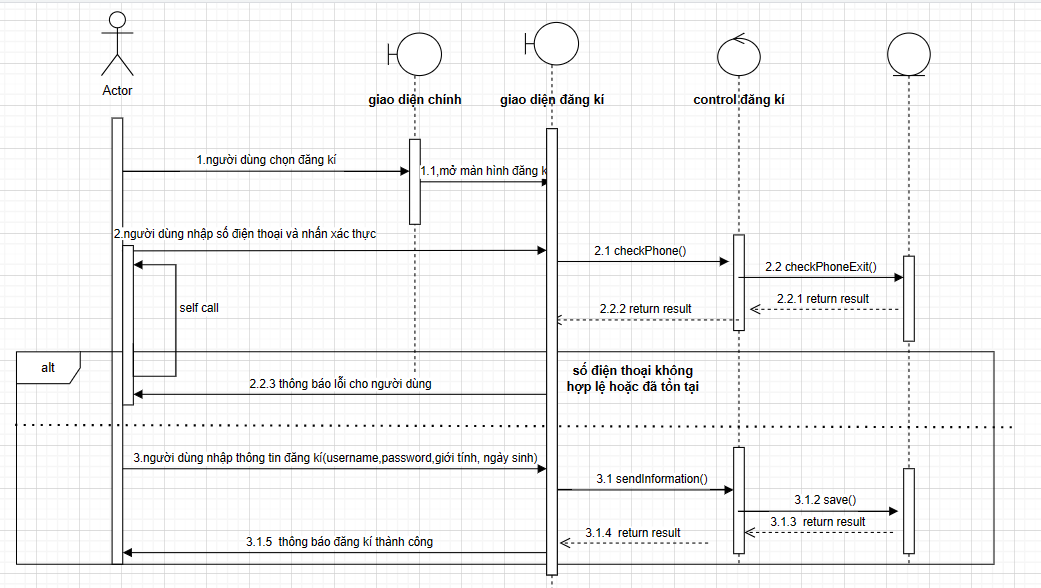


|  |  |  |
| --- | --- | --- |
| **UC002\_Đăng ký học phần mới** | | |
| **Mục đích** | Cho phép người dùng đăng kí học phần | |
| **Mô tả** | Usecase này mô tả cách người dùng đăng kí học phần mới | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Phải đăng ký học môn tiên quyết trước | |
| **Điều kiện sau** | Người dùng đăng kí học phần thành công | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn mục đăng kí học phần | | 2. Hệ thống hiển thị trang đăng ký học phần |
| 3. Người dùng chọn học kỳ và tích vào ô học mới, học lại hoặc học cải thiện | | 4. Hệ thống hiển thị môn học/ học phần đang chờ đăng ký |
| 5. Người dùng chọn vào môn học cần đăng ký | | 6. Hệ thống hiển thị chi tiết lớp học phần |
| 7. Người dùng chọn lớp học phần và nhóm thực hành | | 8.Hệ thống hiển thị thông tin trạng thái, nhóm & sĩ số |
| 9.Người dùng bấm đăng ký | | 10. Hệ thống kiểm tra sĩ số lớp học phần |
|  | | 11.Hệ thống thông báo đăng ký thành công |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |
|  | | 10.1 Nếu lớp học phần bị đầy, hệ thống sẽ thông báo lớp học phần này đã đủ số lượng học viên, không được phép đăng ký vào lớp này**.** |
| 10.2 Người dùng quay về bước 6 | |  |

Activity diagram.

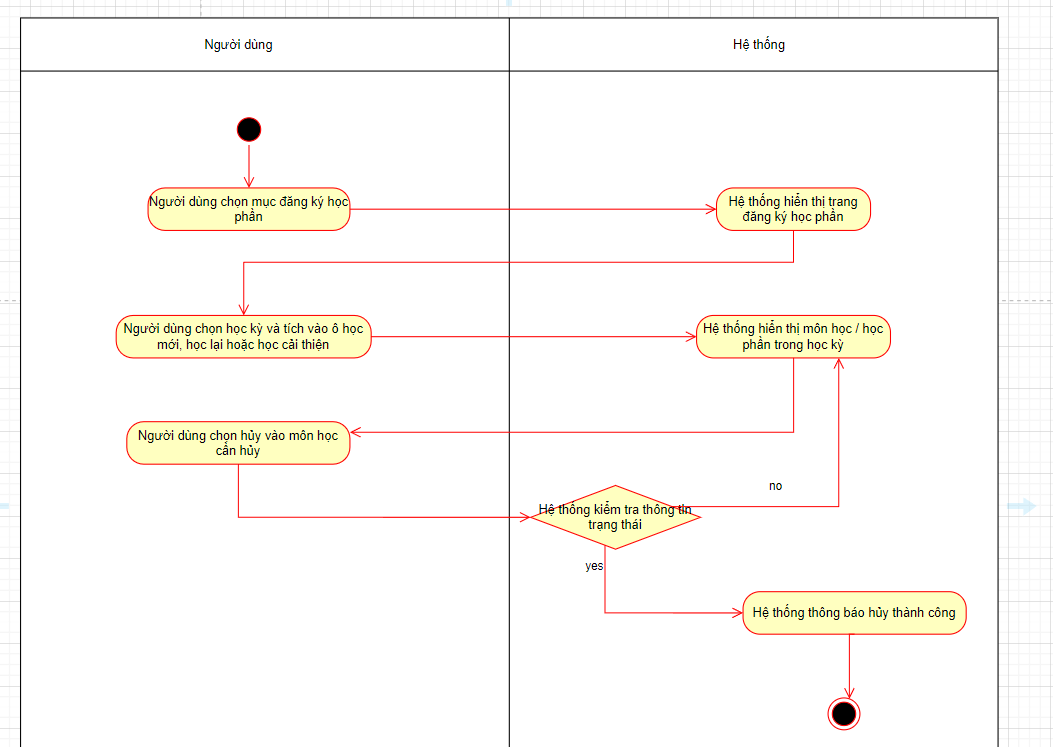


Sequence diagram cho từng tình huống.

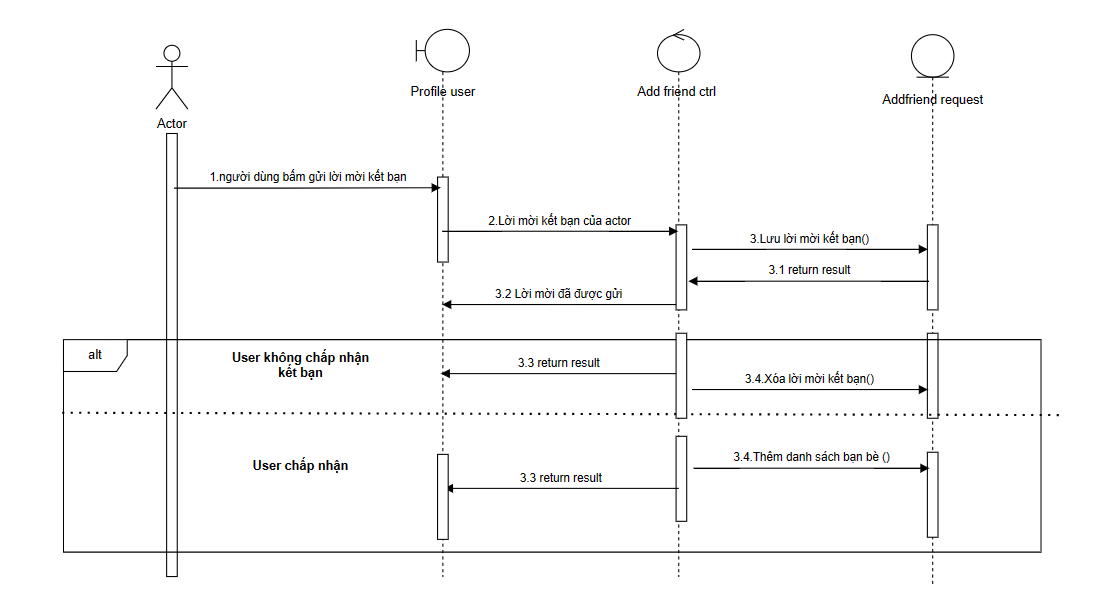


|  |  |  |
| --- | --- | --- |
| **UC003\_Hủy học phần** | | |
| **Mục đích** | Cho phép người dùng hủy học phần vừa đăng ký | |
| **Mô tả** | Usecase này mô tả cách người dùng hủy học phần vừa đăng ký | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đang đăng nhập tài khoản cá nhân | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn mục đăng kí học phần | | 2. Hệ thống hiển thị trang đăng ký học phần |
| 3. Người dùng chọn học kỳ và tích vào ô học mới, học lại hoặc học cải thiện | | 4. Hệ thống hiển thị môn học/ học phần đang chờ đăng ký |
| 5. Người dùng chọn hủy vào môn học cần hủy | | 6. Hệ thống kiểm tra thông tin trạng thái |
|  | | 7.Hệ thống thông báo hủy thành công |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |
|  | | 7.1 Nếu trạng thái lớp học phần đã khóa, hiển thị thông báo hủy thất bại |
| 7.2 Người dùng quay về bước 4 | |  |

Activity diagram.

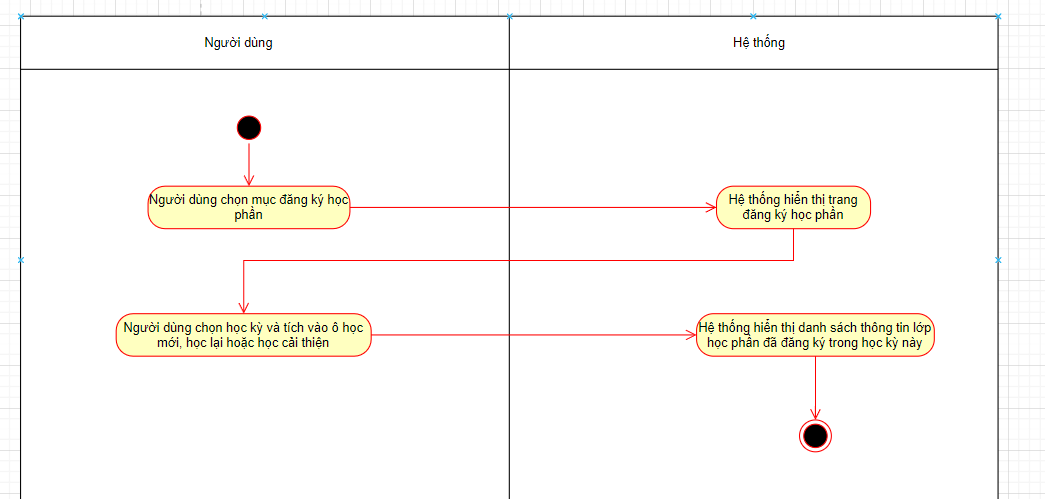


Sequence diagram cho từng tình huống.

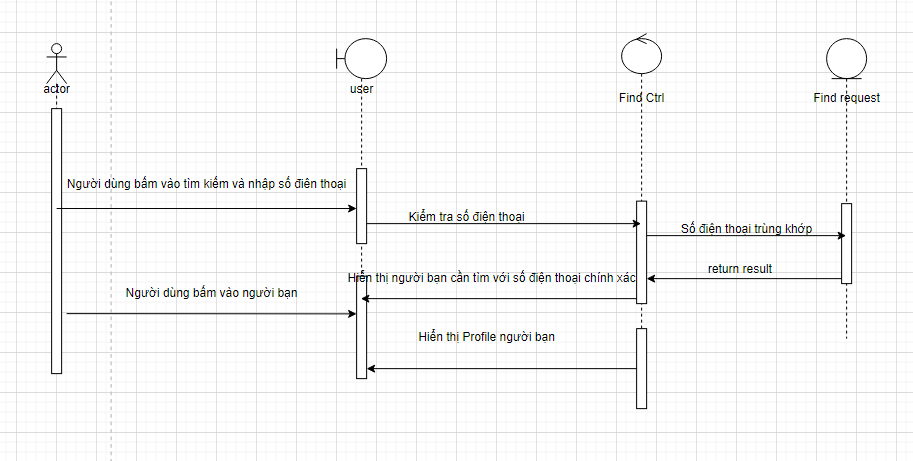


|  |  |  |
| --- | --- | --- |
| **UC004\_Xem danh sách môn học đã đăng ký trong học kỳ** | | |
| **Mục đích** | Cho phép người dùng xem danh sách môn học đã đăng ký trong học kỳ | |
| **Mô tả** | Usecase này mô tả cách người dùng xem danh sách môn học đã đăng ký trong học kỳ | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đang đăng nhập tài khoản cá nhân | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn mục đăng kí học phần | | 2. Hệ thống hiển thị trang đăng ký học phần |
| 3. Người dùng chọn học kỳ và tích vào ô học mới, học lại hoặc học cải thiện | | 4. Hệ thống hiển thị danh sách thông tin lớp học phần đã đăng ký trong học kỳ này |
| **Luồng sự kiện phụ** | |  |
| **Actor** | | **System** |
|  | |  |

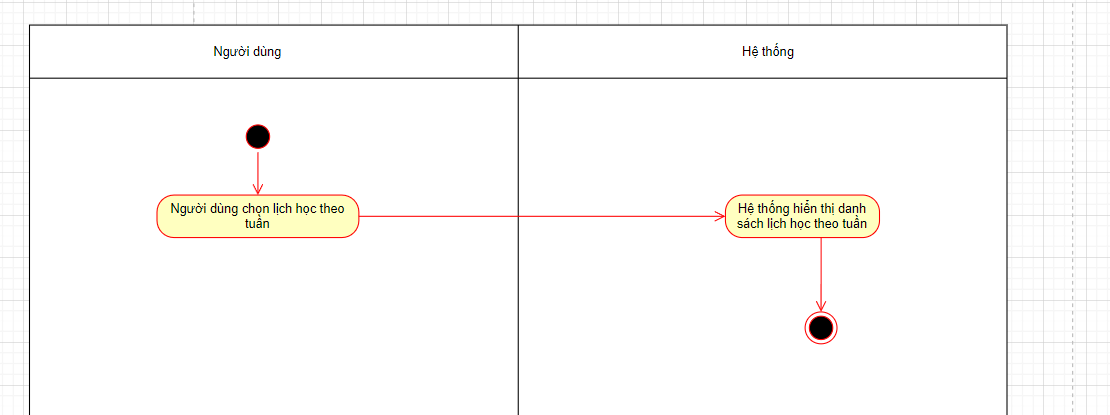
Activity diagram.



Sequence diagram cho từng tình huống.0

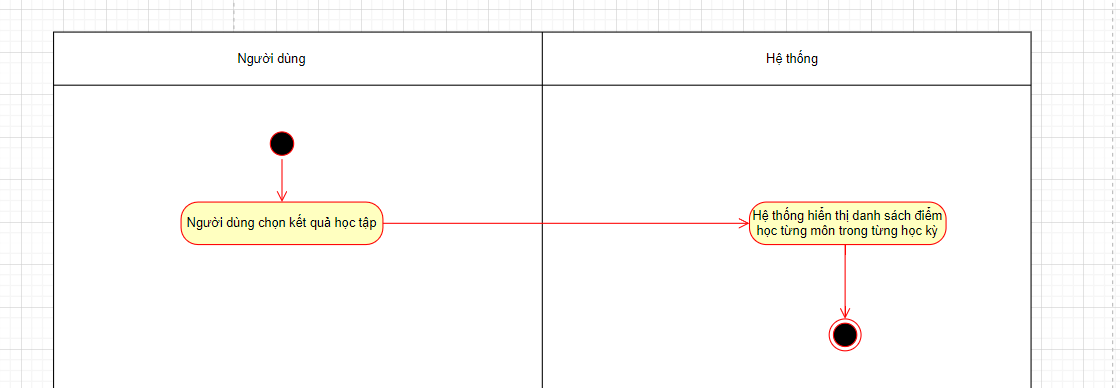


|  |  |  |
| --- | --- | --- |
| **UC005\_Xem lịch học** | | |
| **Mục đích** | Cho phép người dùng xem lịch học | |
| **Mô tả** | Usecase này mô tả cách người dùng xem lịch học | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đang đăng nhập tài khoản cá nhân | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn lịch học theo tuần | | 2. Hệ thống hiện thị danh sách lịch học theo tuần |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |



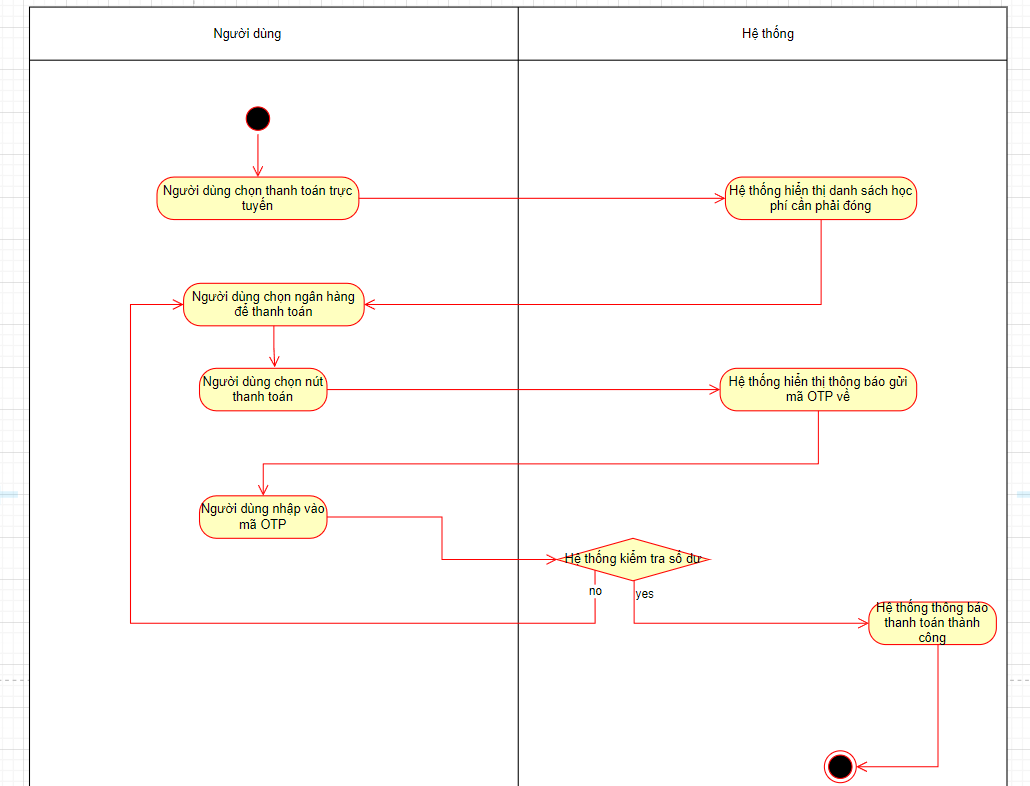
|  |  |  |
| --- | --- | --- |
| **UC006\_Xem điểm** | | |
| **Mục đích** | Cho phép người dùng xem điểm | |
| **Mô tả** | Usecase này mô tả cách người dùng xem điểm cá nhân | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đang đăng nhập tài khoản cá nhân | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng bấm chọn kết quả học tập | | 2. Hệ thống hiển thị danh sách điểm học trong từng học kỳ |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |

* Sơ đồ Activiti:



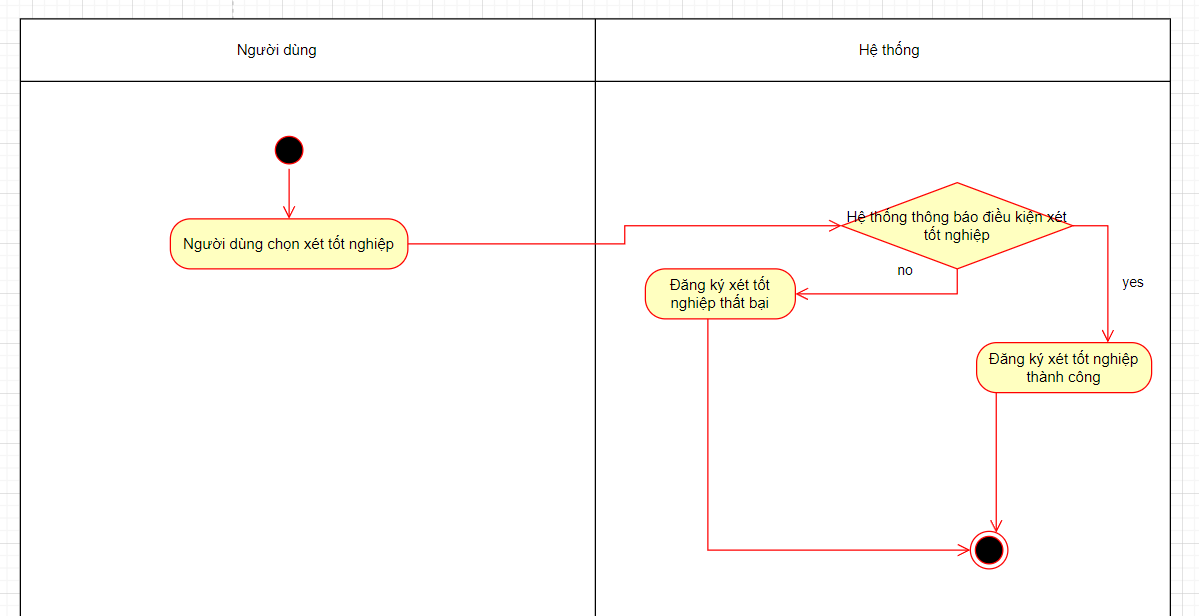
|  |  |  |
| --- | --- | --- |
| **UC007\_Đóng học phí Onlline** | | |
| **Mục đích** | Cho phép người dùng thực hiện đóng học phí trực tuyến | |
| **Mô tả** | Usecase này mô tả cách người dùng đóng học phí trực tuyến | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đang đăng nhập tài khoản cá nhân | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng bấm chọn thanh toán trực tuyến | | 2. Hệ thống hiện thị danh sách các khoản học phí cần phải đóng |
| 3. Người dùng chọn ngân hàng để thanh toán | |  |
| 4. Người dùng chọn nút thanh toán | | 5. Hệ thống hiển thị thông báo gửi mã OTP về |
| 6. Người dùng nhập vào mã OTP | | 7. Hệ thống kiểm tra số dư |
|  | | 8. Hệ thống thông báo thanh toán thành công |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |
|  | | 7.1 Nếu số dư không đủ thì sẽ thông báo thanh toán thất bại |
| 7.2 Người dùng quay về bước 3 | |  |

* Sơ đồ Activiti:



|  |  |  |
| --- | --- | --- |
| **UC008\_** Đăng ký xét tốt nghiệp | | |
| **Mục đích** | Cho phép người dùng đăng ký xét tốt nghiệp | |
| **Mô tả** | Usecase này mô tả cách người dùng xét tốt nghiệp | |
| **Tác nhân** | Người dùng | |
| **Điều kiện trước** | Người dùng đã có tài khoản & đủ điều kiện xét tốt nghiệp | |
| **Điều kiện sau** | Không | |
| **Luồng sự kiện chính** | | |
| **Actor** | | **System** |
| 1. Người dùng chọn mục xét tốt nghiệp | | 2. Hệ thống thông báo đủ điều xét tốt nghiệp |
| **Luồng sự kiện phụ** | | |
| **Actor** | | **System** |
|  | | 2.1 Nếu không đủ điều kiện xét, hệ thống thông báo đăng ký xét tốt nghiệp thất bại |

* Sơ đồ Activiti:



## Class diagram

Mô hình lớp (Mô hình lớp đầy đủ + Mô hình lớp bao gồm các lớp Entity)

## Deployment diagram

Mô hình triển khai hệ thống.

# : HIỆN THỰC

## Cấu hình phần cứng, phần mềm

Ghi rõ thông tin sử dụng các dịch vụ nào của AWS để thực hiện đồ án.

Cấu hình phần cứng, phần mềm của Server (nếu có).

Cấu hình phần cứng, phần mềm của Client trong hệ thống (nếu có).

## Giao diện của hệ thống

Màn hình chức năng đã thực hiện.

## Kế hoạch và hiện thực kiểm thử hệ thống

### Kế hoạch kiểm thử

### Kiểm thử hệ thống

# : KẾT LUẬN

## Kết quả đạt được

1. Đăng nhập và Đăng ký:

• Đăng nhập: Cho phép người dùng nhập thông tin đăng nhập (số điện thoại và mật khẩu) để truy cập vào tài khoản của họ.

• Đăng ký: Cho phép người dùng tạo mới tài khoản bằng cách cung cấp thông tin cần thiết như tên người dùng, số điện thoại và mật khẩu, sau đó xác nhận thông tin qua số điện thoại.

2. Quên Mật khẩu:

• Quên Mật khẩu: Cung cấp cho người dùng một cách để khôi phục mật khẩu nếu họ quên mật khẩu của mình.

• Gửi số điện thoại khôi Phục: Sau khi người dùng yêu cầu khôi phục mật khẩu, hệ thống sẽ gửi một số điện thoại hoặc tin nhắn văn bản chứa một liên kết hoặc mã OTP để họ có thể đặt lại mật khẩu của mình.

3. OTP (One-Time Password):

• Xác thực Hai Bước: Sử dụng OTP để tăng cường tính bảo mật khi đăng nhập hoặc thực hiện các thao tác quan trọng như đặt lại mật khẩu.

• Gửi và Xác Nhận OTP: Hệ thống sẽ gửi một mã OTP duy nhất đến số điện thoại của người dùng, và họ sẽ cần nhập mã này vào ứng dụng để xác nhận danh tính của họ.

4. Chức năng chat đơn:

• Gửi và nhận tin nhắn: Người dùng có thể gửi và nhận tin nhắn với một người khác một cách thành công.

• Thiết lập kết nối: Hệ thống phải có khả năng thiết lập kết nối giữa các người dùng để họ có thể trò chuyện với nhau.

• Bảo mật: Đảm bảo tính bảo mật của các cuộc trò chuyện riêng tư, bằng cách mã hóa dữ liệu và cung cấp cơ chế xác thực.

5. Chức năng chat nhóm:

• Tạo và quản lý nhóm: Người dùng có thể tạo và quản lý các nhóm chat, thêm hoặc loại bỏ thành viên, chỉnh sửa thông tin nhóm, v.v.

• Gửi tin nhắn trong nhóm: Thành viên của nhóm có thể gửi tin nhắn trong nhóm chat, mà tất cả mọi người trong nhóm có thể thấy.

• Thông báo: Cung cấp tính năng thông báo để thông báo cho người dùng khi có tin nhắn mới trong nhóm của họ.

6. Giao diện người dùng:

• Thân thiện và dễ sử dụng: Giao diện người dùng phải thân thiện, dễ sử dụng và dễ dàng để điều hướng.

• Tích hợp các tính năng khác: Cung cấp các tính năng như chụp ảnh, gửi hình ảnh, video, biểu tượng cảm xúc, v.v., để tăng tính đa dạng và tính tương tác.

7. Tính linh hoạt và mở rộng:

• Hỗ trợ đa nền tảng: Ứng dụng chat có thể hoạt động trên nhiều nền tảng, bao gồm web, di động và máy tính để bàn.

• Dễ dàng mở rộng: Hệ thống có thể mở rộng dễ dàng để chịu được lượng người dùng tăng cao mà không làm giảm hiệu suất hoặc chất lượng dịch vụ.

8. Thử nghiệm và sửa lỗi:

• Kiểm thử chất lượng: Thực hiện kiểm thử chất lượng kỹ lưỡng để đảm bảo tính ổn định và hiệu suất của ứng dụng.

• Sửa lỗi và cải thiện: Liên tục sửa lỗi và cải thiện ứng dụng dựa trên phản hồi từ người dùng để cung cấp trải nghiệm người dùng tốt nhất.

## Hạn chế của đồ án

* Bảo mật:

Rủi ro an ninh: Các ứng dụng chat có thể bị tấn công từ các hacker hoặc phần mềm độc hại, đặc biệt khi không có các biện pháp bảo mật đủ mạnh.

Rủi ro bảo mật dữ liệu: Dữ liệu cá nhân và tin nhắn có thể bị đánh cắp hoặc rò rỉ nếu không có cơ chế bảo vệ thích hợp.

* Quản lý tài nguyên:

Tài nguyên máy chủ: Nếu ứng dụng chat trở nên phổ biến, việc quản lý tài nguyên máy chủ có thể trở thành một vấn đề, đặc biệt là khi phải xử lý hàng triệu người dùng cùng một lúc.

Băng thông: Số lượng lớn người dùng có thể tạo áp lực lớn lên băng thông, đặc biệt là khi họ gửi hoặc nhận các tệp tin lớn như hình ảnh hoặc video.

* Hỗ trợ khách hàng:

Xử lý khiếu nại: Có thể xuất hiện nhiều khiếu nại từ người dùng về các vấn đề như lỗi ứng dụng, vấn đề kết nối hoặc mất mát dữ liệu, đòi hỏi một hệ thống hỗ trợ khách hàng mạnh mẽ.

* Tích hợp và tương thích:

Tương thích thiết bị: Đồ án phải hoạt động trên nhiều nền tảng và thiết bị khác nhau (di động, máy tính bảng, máy tính để bàn), và đảm bảo tính tương thích với các phiên bản phần cứng và phần mềm khác nhau.

Tích hợp với các dịch vụ bên thứ ba: Cần tích hợp với các dịch vụ như hệ thống đăng nhập xã hội hoặc dịch vụ lưu trữ đám mây, điều này có thể tạo ra thách thức trong việc quản lý tích hợp và bảo trì.

* Quản lý dữ liệu:

Quản lý dữ liệu lớn: Việc lưu trữ và quản lý lượng lớn dữ liệu từ hàng triệu người dùng có thể đặt ra thách thức về hiệu suất và chi phí.

Tuân thủ quy định: Đảm bảo rằng dự án tuân thủ các quy định và quy định pháp luật liên quan đến bảo vệ dữ liệu cá nhân và quản lý dữ liệu.

## Hướng phát triển

* Nâng cao tính bảo mật:

Mã hóa end-to-end: Cung cấp tính năng mã hóa end-to-end để đảm bảo rằng chỉ người gửi và người nhận có thể đọc tin nhắn.

Xác thực hai bước: Tích hợp xác thực hai bước để cung cấp một lớp bảo mật bổ sung cho người dùng.

Mở rộng tính năng:

Gọi video và gọi thoại: Thêm tính năng gọi video và gọi thoại để mở rộng khả năng giao tiếp của ứng dụng.

Chia sẻ màn hình: Cho phép người dùng chia sẻ màn hình của họ trong các cuộc trò chuyện nhóm hoặc cá nhân.

* Tích hợp trí tuệ nhân tạo (AI) và chatbot:

Chatbot: Phát triển chatbot để cung cấp hỗ trợ tự động cho người dùng, như trả lời câu hỏi phổ biến hoặc hướng dẫn sử dụng.

Phân tích dữ liệu: Sử dụng trí tuệ nhân tạo để phân tích dữ liệu từ cuộc trò chuyện và cung cấp gợi ý hoặc chức năng cá nhân hóa cho người dùng.

* Tích hợp với các nền tảng và dịch vụ khác:

Tích hợp xã hội: Cho phép người dùng đăng nhập bằng tài khoản xã hội và chia sẻ nội dung từ ứng dụng chat lên các mạng xã hội.

Tích hợp dịch vụ bên thứ ba: Cung cấp tích hợp với các dịch vụ bên thứ ba như Google Drive, Dropbox để người dùng có thể chia sẻ và lưu trữ tệp tin dễ dàng.

Tối ưu hóa hiệu suất và kinh nghiệm người dùng:

Tối ưu hóa dữ liệu: Tối ưu hóa việc truyền dữ liệu để cải thiện hiệu suất và tiết kiệm băng thông.

Tối ưu hóa giao diện người dùng: Cải thiện giao diện người dùng để tạo ra trải nghiệm trực quan và dễ sử dụng hơn.

* Mở rộng nền tảng và tích hợp thanh toán:

Tích hợp thanh toán: Cho phép người dùng thanh toán trực tiếp từ ứng dụng để mua các dịch vụ hoặc chức năng bổ sung.

Phát triển API mở: Phát triển API mở để cho phép các nhà phát triển bên thứ ba tạo ra các ứng dụng hoặc tích hợp với ứng dụng chat của bạn.

# TÀI LIỆU THAM KHẢO

Các tài liệu Tiếng Anh

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

Các tài liệu từ Internet

1. <https://nodejs.org/en/docs/>
2. <https://expressjs.com/en/guide/routing.html>