**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**



**ĐỀ TÀI MÔN CÔNG NGHỆ MỚI**

**xây dựng app chat zalo**

***Nhóm 12 - Sinh viên thực hiện***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***STT*** | ***Họ tên sinh viên*** | ***MSSV*** | ***Lớp*** | ***Ghi chú*** |
| 1 | Ngô Nhật Thái | 20047201 | DHKTPM16A | Nhóm trưởng |
| 2 | Nguyễn Hồng Sơn | 20051041 | DHKTPM17C |  |
| 3 | Trần Minh Thuận | 21138281 | DHKTPM16A |  |
| 4 | Ngô Ngọc Vũ | 20047731 | DHKTPM16A |  |

# MỤC LỤC

[MỤC LỤC 1](#_Toc167046707)

[DANH MỤC CÁC HÌNH VẼ 3](#_Toc167046708)

[DANH MỤC CÁC BẢNG BIỂU 4](#_Toc167046709)

[CHƯƠNG 1 : GIỚI THIỆU 5](#_Toc167046710)

[1.1 Tổng quan 5](#_Toc167046711)

[1.2 Mục tiêu đề tài 5](#_Toc167046712)

[1.3 Phạm vi đề tài 5](#_Toc167046713)

[1.4 Mô tả yêu cầu chức năng 5](#_Toc167046714)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 7](#_Toc167046715)

[2.1 Node.js 7](#_Toc167046716)

[2.2 MongoDB 7](#_Toc167046717)

[2.3 Dịch vụ chứng thực Firebase 8](#_Toc167046718)

[2.4 Thư viện Fontend React 9](#_Toc167046719)

[CHƯƠNG 3 KIẾN TRÚC PHẦN MỀM 10](#_Toc167046720)

[3.1 Mô tả kiến trúc MVC với Node.js 10](#_Toc167046721)

[3.1.1 Model 10](#_Toc167046722)

[3.1.2 View 10](#_Toc167046723)

[3.1.3 Controller 10](#_Toc167046724)

[3.2 Lý do chọn kiến trúc 11](#_Toc167046725)

[3.3 Thiết kế kiến trúc 12](#_Toc167046726)

[3.4 Hiện thực kiến trúc 13](#_Toc167046727)

[CHƯƠNG 4 : PHÂN TÍCH VÀ THIẾT KẾ 15](#_Toc167046728)

[4.1 Phân tích yêu cầu bằng UML 15](#_Toc167046729)

[4.1.1 Usecase tổng quát 15](#_Toc167046730)

[4.1.2 Danh sách tác nhân và mô tả 15](#_Toc167046731)

[4.1.3 Danh sách các tình huống hoạt động (Use cases) 16](#_Toc167046732)

[4.2 Class diagram 23](#_Toc167046733)

[4.2.1 Mô hình lớp tổng quát 23](#_Toc167046734)

[4.2.2 Mô hình lớp Entity 23](#_Toc167046735)

[4.3 Deployment diagram 23](#_Toc167046736)

[CHƯƠNG 5 : HIỆN THỰC 24](#_Toc167046737)

[5.1 Cấu hình phần cứng, phần mềm 24](#_Toc167046738)

[5.2 Giao diện của hệ thống (web) 24](#_Toc167046739)

[5.2.1 Login 24](#_Toc167046740)

[5.2.2 Giao diện Home 25](#_Toc167046741)

[5.2.3 Giao diện Đăng ký học phần 25](#_Toc167046742)

[5.2.4 Màn hình lịch học 27](#_Toc167046743)

[5.2.5 Đổi mật khẩu 27](#_Toc167046744)

[5.2.6 Quản lý lớp học phần 28](#_Toc167046745)

[5.2.7 Giao diện quản lý sinh viên của lớp 28](#_Toc167046746)

[5.3 Kế hoạch và hiện thực kiểm thử hệ thống 29](#_Toc167046747)

[5.3.1 Kế hoạch kiểm thử 29](#_Toc167046748)

[5.3.2 Kiểm thử hệ thống 30](#_Toc167046749)

[CHƯƠNG 6 : KẾT LUẬN 32](#_Toc167046750)

[6.1 Kết quả đạt được 32](#_Toc167046751)

[6.2 Hạn chế của đồ án 32](#_Toc167046752)

[6.3 Hướng phát triển 32](#_Toc167046753)

[TÀI LIỆU THAM KHẢO 33](#_Toc167046754)

# DANH MỤC CÁC HÌNH VẼ

[Hình 2‑1 Node.js development 7](#_Toc167046755)

[Hình 2‑2 Mongo Cloud 7](#_Toc167046756)

[Hình 2‑3 Firebase Authencation 8](#_Toc167046757)

[Hình 2‑4 React Library 9](#_Toc167046758)

[Hình 3‑1 Mô hình MVC 10](#_Toc167046759)

[Hình 3‑2 Mô hình MVC tổng quát 12](#_Toc167046760)

[Hình 3‑3 Mô hình MVC – thiết kế 13](#_Toc167046761)

[Hình 3‑4 Hiện thực mô hình lên dự án Node.js 13](#_Toc167046762)

[Hình 5‑1. Giao diện đăng nhập 24](#_Toc167046763)

[Hình 5‑2 Giao diện trang chủ 25](#_Toc167046764)

[Hình 5‑3 Giao diện đăng ký học phần 25](#_Toc167046765)

[Hình 5‑4 Giao diện lịch học 27](#_Toc167046766)

[Hình 5‑5 Giao diện đổi mật khẩu 27](#_Toc167046767)

[Hình 5‑6 Giao diện quản lý lớp học phần 28](#_Toc167046768)

[Hình 5‑7 Quản lý sinh viên lớp 28](#_Toc167046769)

# 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)

# : GIỚI THIỆU

## Tổng quan

Dự án ***Đăng ký Học phần*** là một nền tảng trực tuyến được phát triển nhằm tạo ra phục vụ việc đăng ký và quản lý học phần của sinh viên. Với Dự án Đăng ký Học phần, sinh viên có thể dễ dàng đăng ký các học phần, xem lịch học, và xem tiến trình học của mình từ bất kỳ thiết bị nào có kết nối internet.

## Mục tiêu đề tài

Mục tiêu của dự án là xây dựng một trang web với các tính năng tiện dụng và trực quan đáp ứng nhu cầu đăng kí và quản lý học phần của sinh viên. Đồng thời, nhóm 6 cũng muốn tạo ra một trải nghiệm người dùng tốt nhất có thể, với giao diện thân thiện và tính bảo mật cao.

## Phạm vi đề tài

Phạm vi của dự án bao gồm:

1. Phát triển dự án trên nền tảng web phục vụ cho các thiết bị có kết nối Internet.

2. Cung cấp tính năng về quản lý tài khoản, thông tin học tập và đăng ký học phần.

3. Đảm bảo tính bảo mật và quyền riêng tư cho người dùng.

4. Tích hợp các tính năng phụ như xem lịch học, xem kết quả học tập.

## Mô tả yêu cầu chức năng

***Dự án Đăng ký học phần có các yêu cầu chức năng sau:***

1. Tạo tài khoản/ đăng nhập/ quản lý mật khẩu

2. Xem các môn học theo chuyên ngành, xem các lớp học tương ứng theo môn và đăng ký.

3. Xác nhận các lớp đã đăng ký.

4. Xem lại lịch học đã đăng ký.

5. Xem kết quả học tập.

6. Xem thông tin chung trên trang cá nhân sinh viên.

***Yêu cầu phi chức năng:***

***1. Tính khả dụng:*** Ứng dụng phải có thời gian hoạt động cao, không gặp sự cố đột ngột và có khả năng xử lý tải cao cũng như đảm bảo trải nghiệm mượt mà và không giật lag cho người dùng.

***2. Tính bảo mật:*** Đảm bảo an toàn cho dữ liệu cá nhân của người dùng bằng cách sử dụng mã hóa và các biện pháp bảo mật phù hợp.

***3. Dễ sử dụng:*** Ứng dụng phải có giao diện người dùng thân thiện, dễ sử dụng và dễ dàng hiểu, không gây nhầm lẫn cho người dùng mới.

***4. Tính tương thích:*** Ứng dụng phải tương thích với nhiều loại thiết bị và hệ điều hành khác nhau, bao gồm cả máy tính, điện thoại di động và máy tính bảng.

# : CƠ SỞ LÝ THUYẾT

## Node.js

Hình 2‑1 Node.js development

**Node.js Development:** Node.js được sử dụng làm nền tảng phát triển chính cho dự án Zalo App Chat. Node.js cung cấp một môi trường thực thi JavaScript hiệu quả cho phía máy chủ, cho phép ứng dụng xây dựng trên cơ sở mã nguồn mở này có thể xử lý hàng nghìn kết nối đồng thời một cách hiệu quả.

Sử dụng Node.js để xây dựng các ứng dụng máy chủ với mã nguồn mở, sử dụng JavaScript làm ngôn ngữ lập trình chính.

Sử dụng Express.js, một khung (framework) phổ biến cho Node.js, để xây dựng các API RESTful cho việc tương tác giữa máy chủ và ứng dụng di động.

## MongoDB

Hình 2‑2 Mongo Cloud

**Hệ quản trị cơ sở dữ liệu NoSQL MongoDB:** được tích hợp thông qua dịch vụ Mongo Cloud. MongoDB được sử dụng để lưu trữ dữ liệu người dùng, tin nhắn, và các thông tin liên quan khác

Sử dụng dịch vụ MongoDB Atlas trên Mongo Cloud để triển khai và quản lý các cụm cơ sở dữ liệu MongoDB trên đám mây.

Tận dụng tính năng tự động mở rộng và sao lưu dữ liệu để đảm bảo tính sẵn sàng và độ tin cậy cao cho cơ sở dữ liệu.

## Dịch vụ chứng thực Firebase

Hình 2‑3 Firebase Authencation

**Firebase** là một nền tảng phát triển ứng dụng di động và web của Google, cung cấp các dịch vụ đám mây cho việc phát triển ứng dụng nhanh chóng và hiệu quả. Một trong những tính năng phổ biến của Firebase là Firebase Authentication, một dịch vụ cho phép xác thực người dùng trong ứng dụng của bạn thông qua nhiều phương thức, bao gồm xác thực qua số điện thoại (OTP).

Firebase OTP (One-Time Password) là một phương thức xác thực cho phép người dùng đăng nhập vào ứng dụng thông qua việc nhập một mã OTP được gửi đến số điện thoại của họ. Firebase cung cấp các API dễ sử dụng để triển khai và quản lý việc xác thực OTP, bao gồm gửi mã OTP, xác thực mã OTP và quản lý trạng thái xác thực.

Khi kết hợp Firebase OTP với React, có thể xây dựng một trải nghiệm đăng nhập an toàn và dễ sử dụng cho người dùng Zalo App Chat.

## Thư viện Fontend React

Hình 2‑4 React Library

**React** là một lựa chọn lý tưởng cho việc phát triển giao diện người dùng trong dự án Zalo App Chat. Dự án sử dụng React để xây dựng các thành phần giao diện linh hoạt, tái sử dụng và dễ bảo trì.

React cung cấp khả năng quản lý trạng thái với **Redux:** bao gồm trạng thái của quá trình xác thực OTP, người dùng hiện tại của ứng dụng và các thông tin trạng thái, ... Điều này giúp kiểm soát và hiển thị dữ liệu 1 cách logic.

# KIẾN TRÚC PHẦN MỀM

Hình 3‑1 Mô hình MVC

## Mô tả kiến trúc MVC với Node.js

Mô hình MVC (Model-View-Controller) là một kiến trúc phần mềm phổ biến được sử dụng để xây dựng các ứng dụng web. MVC chia ứng dụng thành ba phần chính: Model, View và Controller. Dưới đây là chi tiết về từng thành phần và cách chúng tương tác trong một ứng dụng Node.js

### Model

* Quản lý dữ liệu và logic nghiệp vụ của ứng dụng.
* Thực hiện các hoạt động CRUD (Create, Read, Update, Delete) trên cơ sở dữ liệu.
* Ví dụ: Sử dụng các ORM như Sequelize hoặc Mongoose để tương tác với cơ sở dữ liệu.

### View

* Chịu trách nhiệm hiển thị dữ liệu cho người dùng.
* Tạo giao diện người dùng và phản hồi các yêu cầu của người dùng.
* Ví dụ: Sử dụng các công cụ template như EJS, Pug hoặc Handlebars để tạo các trang HTML động.

### Controller

* Xử lý các yêu cầu từ người dùng và tương tác với Model để lấy dữ liệu cần thiết.
* Chuyển dữ liệu đến View để hiển thị.
* Controller đóng vai trò trung gian giữa Model và View.

## Lý do chọn kiến trúc

***Mô hình MVC*** được chọn cho dự án đăng ký học phần vì tính phân tách logic rõ ràng, khả năng tái sử dụng mã nguồn, linh hoạt trong việc mở rộng và thay đổi, cùng với việc dễ dàng kiểm thử và nâng cấp phần mềm.

**Dự án “Đăng ký học phần” yêu cầu:**

1. Các thành phần hệ thống phân tách rõ ràng, dễ bảo trì và sửa chữa.
2. Đảm bảo hoạt động ổn định, không yêu cầu tốc độ phản hồi cao nhưng cần độ sẵn sàng.
3. Các luồng được cấu hình rõ ràng.
4. Giao diện cần được phân tách để dễ dàng nâng cấp.
5. Khả nâng chịu tải cao.

Do đó, mô hình MVC xây dựng trên Node.js rất thích hợp với các yêu cầu phi chức năng của dự án.

## Thiết kế kiến trúc

A diagram of a computer

Description automatically generated

Hình 3‑2 Mô hình MVC tổng quát

- Luồng Request: Các request từ View được truyền xuống các Routes – sau đó gọi các Controller tương ứng – gọi các Service – thao tác thay đổi dữ liệu với Model (MongoDB).

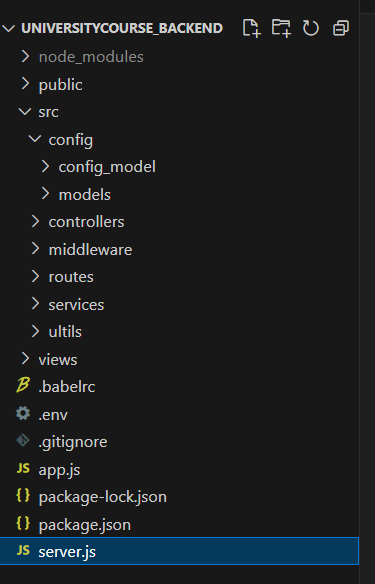
- Luồng Response: Các response từ Model trả lại dữ liệu thông qua luồng Service – Controller – Route. Sau đó được gọi lên UI thông qua Axios để render.

## Hiện thực kiến trúc

A diagram of a computer

Description automatically generated

Hình 3‑3 Mô hình MVC – thiết kế



Hình 3‑4 Hiện thực mô hình lên dự án Node.js

Mô hình triển khai hệ thống

A diagram of a computer network

Description automatically generated

# : PHÂN TÍCH VÀ THIẾT KẾ

## Phân tích yêu cầu bằng UML

### Usecase tổng quát

***A diagram of a diagram

Description automatically generated***

### 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 |
| Sinh viên | Người sử dụng – thao tác với các chức năng của hệ thống |
| Quản lý | Người quản lý các đối tượng trong hệ thống |
| Firebase | Dịch vụ chứng thực tài khoản của người dùng |
| Mongo Cloud | Dịch vụ lưu trữ dữ liệu không quan hệ |

Bảng 3‑1 Danh sách tác nhân và mô tả

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Tên Use case | Mô tả ngắn gọn Use case | Chức năng | Ghi chú |
| *UC01* | Login | Đăng nhập vào trang web | Đăng nhập |  |
| *UC02* | Reset password | Đặt lại mật khẩu cho tài khoản sinh viên | Đặt lại mật khẩu |  |
| *UC03* | Log Personal page | Xem thông tin sinh viên trên trang cá nhân | Xem trang cá nhân |  |
| *UC04* | View subject | Xem các môn học dựa trên chuyên ngành của sinh viên | Xem các môn học |  |
| *UC05* | View Class | Xem các lớp học theo môn học đã chọn | Xem các lớp học |  |
| *UC06* | Register Class | Đăng ký vào lớp học, xác nhận bằng OTP qua qua email sinh viên |  |  |
| *UC07* | View class schedule | Xem lịch các lớp học đã đăng ký | Xem lịch học |  |
| *UC08* | Create course | Tạo ra các môn học | Thêm môn học |  |
| *UC09* | Create class | Thêm mới các lớp học | Thêm lớp học |  |
| *UC10* | Manage student | Quản lý các tài khoản sinh viên | Quản lý tài khoản |  |
| *UC11* | Create User | Thêm mới tài khoản sinh viên | Thêm tài khoản sinh viên |  |

Bảng 3‑2 Danh sách các tình huống trong hệ thống

##### UC01\_Login– Diagram

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| **Use case: UC01\_** Login | | |
| Mục đích: | Đăng nhập | |
| Mô tả: | Đăng nhập vào trang web | |
| Tác nhân: | Sinh viên/ quản lý | |
| Điều kiện trước: | Tài khoản đã tồn tại trong Database của Mongo Cloud | |
| Điều kiện sau: | Trang chuyển đến trang cá nhân sinh viên | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Người dùng nhập thông tin đăng nhập | | 2. Kiểm tra thông tin đăng nhập |
|  | | 3. Chuyển đến trang sinh viên |  |
| − **Luồng sự kiện phụ (exception flow):** | | |  |
|  | | 3.1 Thông báo sai tài khoản |  |
|  | | 3.2 Tải lại trang Login |  |

##### UC02\_ Reset password

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC02\_ Reset password | | |
| Mục đích: | Đặt lại mật khẩu | |
| Mô tả: | Đặt lại mật khẩu cho tài khoản thông qua xác nhận OTP email | |
| Tác nhân: | Sinh viên/ quản lý | |
| Điều kiện trước: | Tài khoản đã tồn tại trong Database của Mongo Cloud | |
| Điều kiện sau: | Mật khẩu mới được đặt lại và lưu vào Mongo Cloud | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Chọn quên mật khẩu | | 2. Hiển thị giao diện quên mật khẩu |
| 3. Nhập mã số sinh viên | | 4. Kiểm tra mã sinh viên để lấy email |  |
|  | | 5. Gửi OTP cho email sinh viên |  |
| 6. Nhập OTP | | 7. Xác nhận OTP thành công |  |
|  | | 8. Cho phép chuyển sang trang tạo mật khẩu |  |
| 9. Nhập và xác nhận mật khẩu mới | | 10. Xác nhận mật khẩu và lưu vào Database |  |
| − **Luồng sự kiện phụ (exception flow):** | | |  |
|  | | 4.1 Báo lỗi không tồn tại |  |

##### UC03\_ Log to Personal page

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC03\_ Log to Personal page | | |
| Mục đích: | Đăng nhập vào trang cá nhân | |
| Mô tả: | Xem thông tin sinh viên trên trang cá nhân | |
| Tác nhân: | Sinh viên/ quản lý | |
| Điều kiện trước: | Đăng nhập thành công | |
| Điều kiện sau: | Không | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
|  | | 1. Hiển thị giao diện cá nhân |
| − **Luồng sự kiện phụ (exception flow):** | | |  |

##### UC04\_ View subject

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC04\_ View subject | | |
| Mục đích: | Xem các môn học | |
| Mô tả: | Xem các môn học dựa trên chuyên ngành của sinh viên | |
| Tác nhân: | Sinh viên/ quản lý | |
| Điều kiện trước: | Đăng nhập thành công, có dữ liệu môn học | |
| Điều kiện sau: | Không | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Chọn đăng ký học phần | | 2. Hiển thị các môn học theo chuyên ngành sinh viên |
| − **Luồng sự kiện phụ (exception flow):** | | |  |

##### UC05\_ View Class

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC05\_ View Class | | |
| Mục đích: | Xem các lớp học | |
| Mô tả: | Xem các lớp học theo môn học đã chọn | |
| Tác nhân: | Sinh viên/ quản lý | |
| Điều kiện trước: | Đăng nhập thành công, có dữ liệu lớp học trong hệ thống | |
| Điều kiện sau: | Không | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Chọn môn học muốn đăng ký | | 2. Hiển thị các lớp tương ứng với môn học |
| − **Luồng sự kiện phụ (exception flow):** | | |  |

##### UC06\_ Register Class

***- Đặc tả Usecase:***

|  |  |  |  |
| --- | --- | --- | --- |
| Use case: UC06\_ Register Class | | | |
| Mục đích: | Đăng ký vào lớp học | | |
| Mô tả: | Đăng ký vào lớp học, xác nhận bằng OTP qua qua email sinh viên | | |
| Tác nhân: | Sinh viên | | |
| Điều kiện trước: | Đăng nhập thành công, lớp còn chỗ trống, không bị trùng lịch | | |
| Điều kiện sau: | Sinh viên được lưu vào danh sách lớp | | |
| − **Luồng sự kiện chính (main flow):** | | | |  |
| **Actor** | | | **System** |
| 1. Chọn lớp muốn đăng ký | | | 2. Hiển thị thông tin lớp |
| 3. Ấn đăng ký | | | 4. Kiểm tra thông tin đăng ký |  |
|  | | | 5. Gửi OTP về email sinh viên |  |
| 6. Nhập OTP xác nhận | | | 7. Kiểm tra OTP |  |
|  | | | 8. Thêm sinh viên vào danh sách lớp |  |
| − **Luồng sự kiện phụ (exception flow):** | | | |  |
|  | | 4.1 Hiển thị không phù hợp | |  |
|  | | 7.1 Hiển thị sai thông tin OTP | |  |

##### UC07\_ View class schedule

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC07\_ View class schedule | | |
| Mục đích: | Xem lịch học | |
| Mô tả: | Xem lịch các lớp học đã đăng ký | |
| Tác nhân: | Sinh viên | |
| Điều kiện trước: | Đã đăng ký các lớp học | |
| Điều kiện sau: | Không | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Chọn lịch học | | 2. Hiển thị lịch học của sinh viên |
| − **Luồng sự kiện phụ (exception flow):** | | |  |

##### UC08\_ Create course

***- Đặc tả Usecase:***

|  |  |  |
| --- | --- | --- |
| Use case: UC08\_ Create course | | |
| Mục đích: | Thêm môn học | |
| Mô tả: | Tạo ra các môn học | |
| Tác nhân: | Quản lý | |
| Điều kiện trước: | Đăng nhập thành công | |
| Điều kiện sau: | Môn học được tạo mới và lưu vào Mongo Cloud | |
| − **Luồng sự kiện chính (main flow):** | | |  |
| **Actor** | | **System** |
| 1. Chọn thêm môn học | | 2. Hiển thị giao diện thêm môn học |
| 3. Nhập các thông tin và ấn Thêm | | 4. Kiểm tra thông tin |  |
|  | | 5. Lưu mới môn học và thông báo thành công |  |
| − **Luồng sự kiện phụ (exception flow):** | | |  |
|  | | 4.1 Thông báo thông tin không phù hợp. |  |

##### UC09\_ Create class

***- Đặc tả Usecase:***

|  |  |  |  |
| --- | --- | --- | --- |
| Use case: UC09\_ Create class | | | |
| Mục đích: | Thêm lớp học | | |
| Mô tả: | Tạo ra các lớp học | | |
| Tác nhân: | Quản lý | | |
| Điều kiện trước: | Đăng nhập thành công | | |
| Điều kiện sau: | Lớp học được tạo mới và lưu vào Mongo Cloud | | |
| − **Luồng sự kiện chính (main flow):** | | | |  |
| **Actor** | | | **System** |
| 1. Chọn thêm môn học | | | 2. Hiển thị giao diện thêm môn học |
| 3. Nhập các thông tin và ấn Thêm | | | 4. Kiểm tra thông tin |  |
|  | | | 5. Lưu mới môn học và thông báo thành công |  |
| − **Luồng sự kiện phụ (exception flow):** | | | |  |
|  | | 4.1 Hiển thị thông báo thông tin không phù hợp | |  |

##### UC11\_ Create User

***- Đặc tả Usecase:***

|  |  |  |  |
| --- | --- | --- | --- |
| Use case: UC11\_ Create User | | | |
| Mục đích: | Thêm tài khoản | | |
| Mô tả: | Tạo ra tài khoản sinh viên mới | | |
| Tác nhân: | Quản lý | | |
| Điều kiện trước: | Đăng nhập thành công | | |
| Điều kiện sau: | Tài khoản sinh viên được tạo mới và lưu vào Mongo Cloud | | |
| − **Luồng sự kiện chính (main flow):** | | | |  |
| **Actor** | | | **System** |
| 1. Chọn thêm tài khoản sinh viên | | | 2. Hiển thị giao diện thêm tài khoản |
| 3. Nhập các thông tin và ấn Thêm | | | 4. Kiểm tra thông tin |  |
|  | | | 5. Lưu mới tài khoản và thông báo thành công |  |
| − **Luồng sự kiện phụ (exception flow):** | | | |  |
|  | | 4.1 Hiển thị thông báo thông tin không phù hợp | |  |

## Class diagram

### Mô hình lớp tổng quát

### Mô hình lớp Entity

## Deployment diagram

Mô hình triển khai hệ thống.

A computer network diagram with blue boxes

Description automatically generated with medium confidence

# : HIỆN THỰC

## Cấu hình phần cứng, phần mềm

Các dịch vụ mà dự án đã sử dụng:

*- Mongo Cloud*

*- Firebase AUTH*

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 (web)

### Login

A screenshot of a computer

Description automatically generated

Hình 5‑1. Giao diện đăng nhập

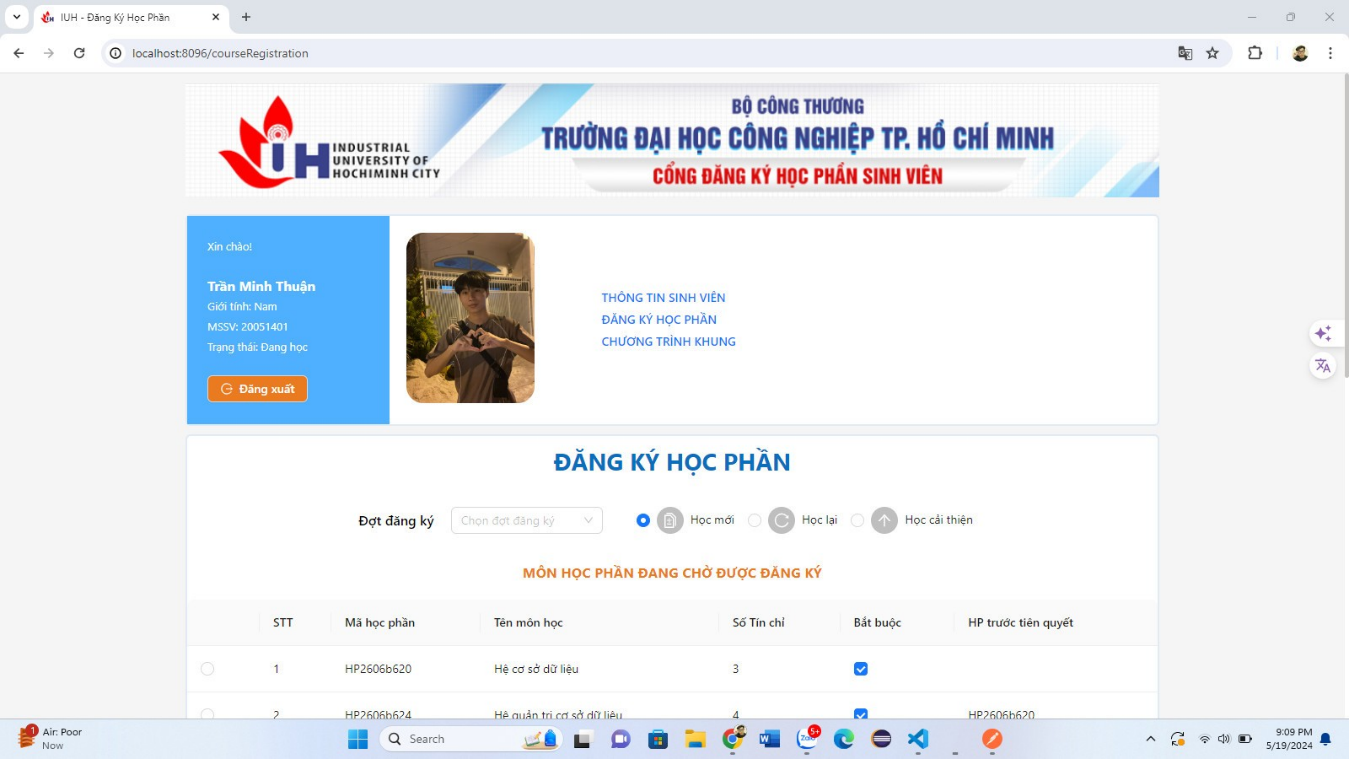
### Giao diện Home

A screenshot of a computer

Description automatically generated

Hình 5‑2 Giao diện trang chủ

### Giao diện Đăng ký học phần



Hình 5‑3 Giao diện đăng ký học phần

A screenshot of a computer

Description automatically generated

**A screenshot of a computer

Description automatically generated**

### Màn hình lịch học

A screenshot of a computer

Description automatically generated

Hình 5‑4 Giao diện lịch học

### Đổi mật khẩu

A screenshot of a computer

Description automatically generated

Hình 5‑5 Giao diện đổi mật khẩu

### Quản lý lớp học phần

A screenshot of a computer

Description automatically generated

Hình 5‑6 Giao diện quản lý lớp học phần

### Giao diện quản lý sinh viên của lớp

A screenshot of a computer

Description automatically generated

Hình 5‑7 Quản lý sinh viên lớp

## Kế hoạch và hiện thực kiểm thử hệ thống

### Kế hoạch kiểm thử

***1. Mục đích kiểm thử:***

Đảm bảo tính ổn định, đáng tin cậy và bảo mật của hệ thống Đăng ký học phần trước khi phát hành.

Xác nhận rằng tất cả các chức năng cơ bản và quan trọng đều hoạt động một cách chính xác và như mong đợi.

Đảm bảo trải nghiệm người dùng suôn sẻ và không gặp phải các lỗi quan trọng trong quá trình sử dụng ứng dụng.

***2. Phạm vi kiểm thử:***

Kiểm thử toàn diện các chức năng cơ bản của hệ thống Đăng ký học phần, bao gồm:

1. Tạo tài khoản/ đăng nhập/ quản lý mật khẩu

2. Xem các môn học theo chuyên ngành, xem các lớp học tương ứng theo môn và đăng ký.

3. Xác nhận các lớp đã đăng ký.

4. Xem lại lịch học đã đăng ký.

5. Xem kết quả học tập.

6. Xem thông tin chung trên trang cá nhân sinh viên.

***3. Phương pháp kiểm thử:***

Sử dụng kỹ thuật kiểm thử tự động và kiểm thử thủ công để đảm bảo tính toàn diện và chất lượng của ứng dụng.

Sử dụng dữ liệu kiểm thử đa dạng để đảm bảo tính khách quan và phủ sóng đầy đủ các trường hợp sử dụng.

Sử dụng các công cụ kiểm thử phần mềm phù hợp như Selenium, Appium, JUnit, và Postman.

Đảm bảo mọi lỗi được ghi lại, báo cáo và ưu tiên để sửa chữa.

***4. Lịch trình kiểm thử:***

Phân chia công việc kiểm thử theo từng chức năng cụ thể, như đã mô tả trong bản kế hoạch chi tiết ngày.

Đảm bảo mỗi phần của kế hoạch kiểm thử được thực hiện một cách có hệ thống và chặt chẽ.

Tính toán thời gian cần thiết cho mỗi phần của kế hoạch và đặt ra các mục tiêu cụ thể cho việc hoàn thành.

***5. Kế hoạch báo cáo:***

Tạo một hệ thống báo cáo để theo dõi tiến độ kiểm thử và ghi lại kết quả kiểm thử.

Báo cáo các lỗi phát hiện, phân loại chúng và ưu tiên để sửa chữa.

Đảm bảo rằng các báo cáo được chia sẻ và phản hồi được đưa ra một cách kịp thời.

***6. Kiểm thử người dùng cuối:***

Tiến hành kiểm thử người dùng cuối để đảm bảo rằng ứng dụng hoạt động một cách ổn định và thoải mái với mọi loại người dùng.

Thu thập phản hồi từ người dùng để cải thiện trải nghiệm và khắc phục các vấn đề cụ thể.

***7. Đánh giá và điều chỉnh:***

Đánh giá kết quả của kiểm thử và điều chỉnh kế hoạch kiểm thử nếu cần thiết để đảm bảo rằng mọi khía cạnh của ứng dụng được kiểm tra một cách đầy đủ và kỹ lưỡng.

### Kiểm thử hệ thống

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***STT*** | ***Test case*** | ***Mô tả*** | ***Thực hiện*** | ***Kết quả mong đợi*** | ***Kết quả thực tế*** | ***Ngày kiểm thử*** | ***Pass/ Fail*** |
| 2 | Đăng nhập | Đăng nhập vào ứng dụng | B1: Chọn đăng nhập và nhập tài khoản mật khẩu  B2: Nhập OTP | Đăng nhập thành công | Đăng nhập thành công | 29/3/2024 | **Pass** |
| 3 | Kết bạn | Gửi lời mời kết bạn | B1: Chọn thêm bạn và nhập số điện thoại  B2: Nhập lời mời và ấn gửi | Gửi lời mời thành công | Gửi lời mời thành công | 5/4/2024 | **Pass** |
| 4 | Chấp nhận kết bạn | Chấp nhận lời mời kết bạn | B1: Chọn mục kết bạn  B2: Chấp nhận lời mời | Hệ thống ghi nhận bạn bè | Hệ thống ghi nhận bạn bè | 5/4/2024 | **Pass** |
| 5 | Thu hồi lời mời | Thu hồi lời mời kết bạn | B1: Chọn bạn đang gửi lời mời  B2: Ấn thu hồi | Hệ thống xóa lời mời 2 bên | Chỉ xóa được bên gửi | 6/4/2024 | **Fail** |
| 6 | Nhắn tin | Gửi tin nhắn | B1: Chọn bạn bè hoặc phòng chat  B2: Soạn tin nhắn và ấn gửi | Hệ thống lưu trữ tin nhắn và đối phương nhận được tin nhắn | Hệ thống lưu trữ tin nhắn và đối phương nhận được tin nhắn | 7/4/2024 | **Pass** |
| 7 | Xóa, thu hồi, ghim, chuyển tiếp | Thao tác với tin nhắn đã gửi | B1: Chọn tin nhắn  B2: Chọn option thao tác | Các thao tác được thực hiện | Các thao tác được thực hiện, trừ chuyển tiếp | 7/4/2024 | **Fail** |
| 8 | Tạo group chat | Tạo mới group chat | B1: Chọn tạo group  B2: Chọn tối thiểu 3 bạn bè và ấn tạo | Group mới được tạo | Group mới được tạo | 12/4/2024 | **Pass** |
| 9 | Thao tác với group | Thao tác thêm, xóa, đổi quyền thành viên cho group | B1: Chọn group  B2: Chọn thao tác và thành viên thực hiện | Thao tác thực hiện thành công | Thao tác thực hiện thành công | 13/4/2024 | **Pass** |

# : KẾT LUẬN

## Kết quả đạt được

Đã hoàn thành đa số các chức năng theo đặc tả dự án, đảm bảo tính hoạt động và tính đúng.

Các chức năng chính đã được kiểm thử kỹ lưỡng và đạt được kết quả tốt.

## Hạn chế của đồ án

Mặc dù đa số các chức năng đã hoàn thành theo đặc tả, nhưng có một số hạn chế nhất định.

Có thể cần thêm thời gian và tài nguyên để xử lý các lỗi và cải thiện hiệu suất của ứng dụng.

Các tính năng mới hoặc cải tiến có thể cần được thêm vào trong tương lai để đáp ứng nhu cầu người dùng và nâng cao khả năng chịu tải của hệ thống.

## Hướng phát triển

Tiếp tục cải thiện và bảo trì ứng dụng để đảm bảo tính ổn định và hiệu suất cao.

Phát triển và triển khai các tính năng mới nhằm nâng cao trải nghiệm người dùng và tạo ra giá trị thêm cho ứng dụng.

Tăng cường bảo mật và quản lý quyền riêng tư để đảm bảo an toàn thông tin của người dùng.

Tăng cường hỗ trợ và tương tác với cộng đồng người dùng để hiểu rõ hơn về nhu cầu và mong muốn của họ và điều chỉnh phát triển dựa trên phản hồi đó.

# 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. [Node.JS Document](https://nodejs.org/en/docs/)
2. [Express Document](https://expressjs.com/en/guide/routing.html)
3. [Firebase Documentation](https://firebase.google.com/docs/): Tài liệu hướng dẫn chi tiết về cách sử dụng Firebase.
4. [MongoDB Atlas Documentation](https://docs.mongodb.com/manual/): Tài liệu chính thức về MongoDB Atlas
5. [Amazon RDS Documentation](https://docs.aws.amazon.com/rds/index.html): Tài liệu chính thức về Amazon RDS
6. [React Documentation](https://legacy.reactjs.org/docs/getting-started.html): Tài liệu hướng dẫn chi tiết về cách sử dụng React
7. [Redux Documentation](https://redux.js.org/introduction/getting-started): Tài liệu về cách sử dụng Redux trong ứng dụng React.
8. [Postman Documentation](https://learning.postman.com/docs/introduction/overview/): Tài liệu về cách sử dụng Postman để kiểm thử API