**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**



**báo cáo tiểu luận cuối khóa**

***Nhóm 05 - Sinh viên thực hiện***

1. Vũ Tiến Đức- 20089731
2. Thái Quang Bảo - 20093191

# MỤC LỤC

[MỤC LỤC 1](#_Toc171947642)

[DANH MỤC CÁC HÌNH VẼ 3](#_Toc171947643)

[DANH MỤC CÁC BẢNG BIỂU 4](#_Toc171947644)

[CHƯƠNG 1 : GIỚI THIỆU 5](#_Toc171947645)

[1.1 Tổng quan 5](#_Toc171947646)

[1.2 Mục tiêu đề tài 5](#_Toc171947647)

[1.3 Phạm vi đề tài 5](#_Toc171947648)

[1.4 Mô tả yêu cầu chức năng 6](#_Toc171947649)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 7](#_Toc171947650)

[2.1 Ngôn ngữ phát triển ứng dụng 7](#_Toc171947651)

[2.1.1 Ngôn ngữ lập trình Java 7](#_Toc171947652)

[2.1.2 Ngôn ngữ lập trình JavaScript 8](#_Toc171947653)

[2.1.3 Ngôn ngữ lập trình TypeScript 9](#_Toc171947654)

[2.2 Những framework hổ trợ phát triển ứng dụng 10](#_Toc171947655)

[2.2.1 Framework Spring boot 10](#_Toc171947656)

[2.2.2 Framework ExpressJS 10](#_Toc171947657)

[2.2.3 Framework NestJS 10](#_Toc171947658)

[2.2.4 Framework NextJS 10](#_Toc171947659)

[2.3 Những công nghệ áp dụng 10](#_Toc171947660)

[2.3.1 Node.js 10](#_Toc171947661)

[2.3.2 MongoDB / DynamoDB 11](#_Toc171947662)

[2.3.3 JWT (Json Token Web) 11](#_Toc171947663)

[2.3.4 Mailer Service 12](#_Toc171947664)

[2.3.5 Session/Cookies 12](#_Toc171947665)

[CHƯƠNG 3 : PHÂN TÍCH VÀ THIẾT KẾ 14](#_Toc171947666)

[3.1 Phân tích yêu cầu bằng UML 14](#_Toc171947667)

[3.1.1 Usecase tổng quát 14](#_Toc171947668)

[3.1.2 Danh sách tác nhân và mô tả 15](#_Toc171947669)

[3.1.3 Danh sách các tình huống hoạt động (Use cases) 15](#_Toc171947670)

[3.1.4 Tình huống hoạt động 17](#_Toc171947671)

[3.2 Class diagram 37](#_Toc171947672)

[3.3 Deployment diagram 37](#_Toc171947673)

[CHƯƠNG 4 : HIỆN THỰC 40](#_Toc171947674)

[4.1 Cấu hình phần cứng, phần mềm 40](#_Toc171947675)

[4.2 Giao diện của hệ thống 40](#_Toc171947676)

[4.3 Kế hoạch và hiện thực kiểm thử hệ thống 60](#_Toc171947677)

[4.3.1 Kế hoạch kiểm thử 60](#_Toc171947678)

[4.3.1.1 Phần cứng: Máy tính cá nhân có kết nối mạng LAN 60](#_Toc171947679)

[4.3.1.2 Phần mềm 61](#_Toc171947680)

[4.3.2 Kiểm thử hệ thống 61](#_Toc171947681)

[CHƯƠNG 5 : KẾT LUẬN 63](#_Toc171947682)

[5.1 Kết quả đạt được 63](#_Toc171947683)

[5.2 Hạn chế của đồ án 63](#_Toc171947684)

[Nhóm chưa làm xong phần admin, video call 63](#_Toc171947685)

[5.3 Hướng phát triển 63](#_Toc171947686)

[TÀI LIỆU THAM KHẢO 65](#_Toc171947687)

# 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)

# : GIỚI THIỆU

## Tổng quan

Dự án Health-care làm một dự án về chăm sóc sức khỏe nhằm tạo ra một hệ thống toàn diện để cung cấp các dịch vụ y tế và chăm sóc sức khỏe hiệu quả. Hệ thống bao gồm các tính năng quản lý thông tin bệnh nhân, đặt lịch hẹn trực tuyến, khám và kê toa thuốc, hỗ trợ hỏi đáp trực tuyến 24/24 thông qua chat GPT, và cung cấp một diễn đàn và cẩm nang y tế để người dùng có thể chia sẻ và tiếp cận các thông tin y tế hữu ích. Bằng việc kết hợp các công nghệ hiện đại và tiên tiến, dự án hướng đến mục tiêu cải thiện quản lý sức khỏe cá nhân và tăng cường sự tiện lợi cho người dùng trong việc truy cập dịch vụ y tế và kiến thức y tế.

## Mục tiêu đề tài

- Mục tiêu của dự án là xây dựng một ứng dụng chăm sóc sức khỏe đa nền tảng (mobile và web) có khả năng cung cấp trải nghiệm giao tiếp thuận tiện và đáng tin cậy cho người dùng, đồng thời đảm bảo tính bảo mật và an toàn thông tin cho dữ liệu của họ.

- Trải nghiệm qui trình trong việc phát triển phần mềm:

+ Tìm hiểu requirement từ khách hàng

+ Design SRS, ADD, DDD

+ Coding hệ thống với Java, NodeJS

+ Tìm hiểu về chatbot, AI

+ Thực hiện testing

+ Deploy hệ thống (cloud computing)

## Phạm vi đề tài

Triển khai một số hệ thống chăm sóc sức khỏe cho các bài toán như:

- Thu thập thông tin bệnh nhân nhằm đưa ra các cảnh báo

- Ứng dụng các API về nhằm xây dựng các ứng dụng ở dạng Microservice + Đặt Lịch hẹn tại các bệnh viện, phòng khám

+ Đặt lịch hẹn tại các bệnh viện, phòng khám online: Bác sĩ và bệnh nhân sẽ vào phòng meet online.

+ Bác sĩ có thể nêu ra tình trạng sức khỏe và kê toa cho bệnh nhân

+ Có phần hỏi đáp nhanh 24/24 sử dụng chat GPT để trực tuyến mọi lúc, giải đáp mọi thắc mắt cho bệnh nhân.

+ Forum để môi người có thể đặt câu hỏi và chia sẻ

+ Có cẩm nang y tế (đa phần là các bài báo và y tế và chăm sóc sức khỏe)

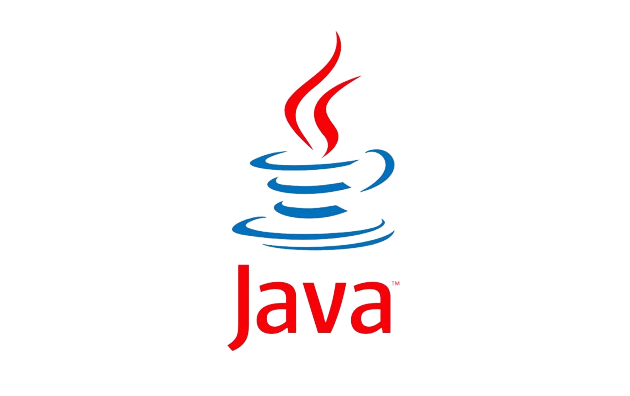
## Mô tả yêu cầu chức năng

# : CƠ SỞ LÝ THUYẾT

## Ngôn ngữ phát triển ứng dụng

### Ngôn ngữ lập trình Java

Java là ngôn ngữ lập trình hướng đối tượng được duy trì và phát triển bởi Oracle. Nó được sử dụng rộng rãi trong phát triển ứng dụng desktop, web, di động và game. Java cho phép phát triển ứng dụng chạy trên mọi nền tảng và có một cộng đồng lập trình viên ngày càng lớn mạnh. Java bao gồm một công cụ thực thi, trình biên dịch và một bộ thư viện phong phú.



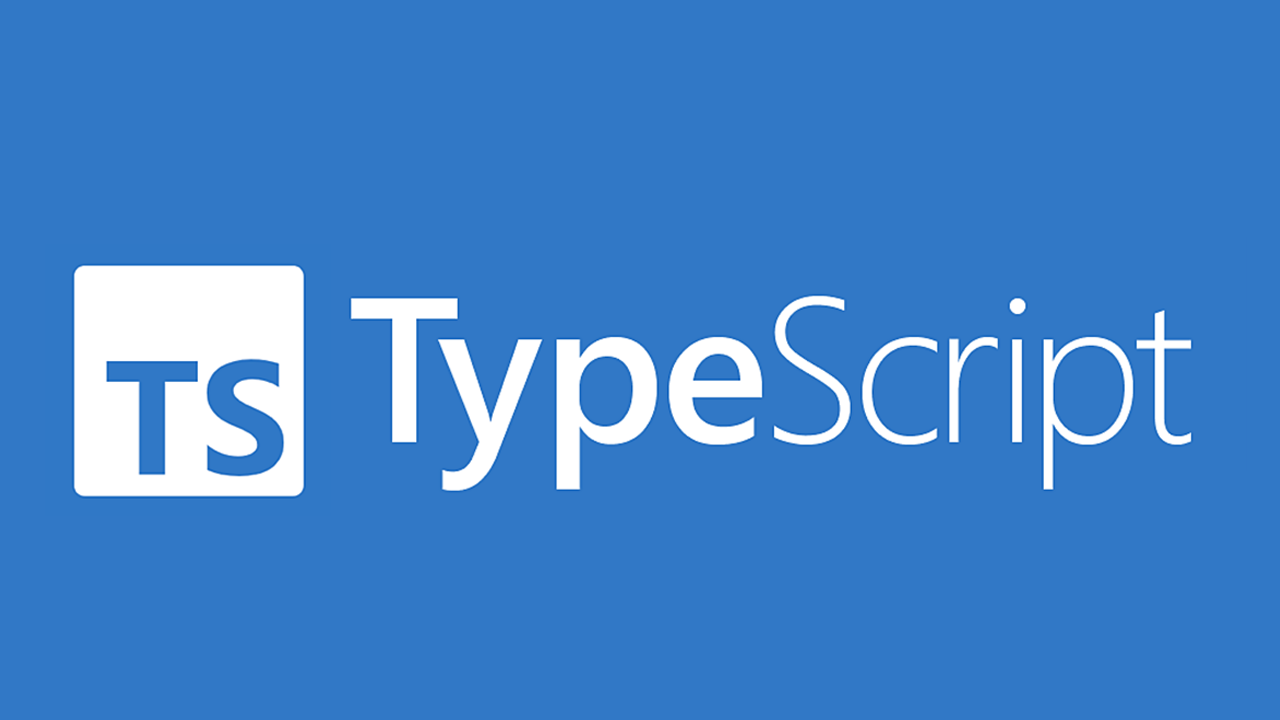
### Ngôn ngữ lập trình JavaScript

JavaScript ra đời vào năm 1995 ban đầu chỉ là một ngôn ngữ kịch bản (scripting language) được sử dụng để tương tác trên các trang web. Với sự phát triển của các công nghệ web và sự xuất hiện của Node.js vào năm 2009, JavaScript đã không còn giới hạn trong việc thực thi trên trình duyệt mà mở rộng sang môi trường máy chủ. Ngày nay, JavaScript được công nhận là một ngôn ngữ lập trình chính thống, hoạt động trên nhiều nền tảng như web và máy chủ. JavaScript dễ học và được sử dụng rộng rãi cho việc phát triển các ứng dụng web, mobile và nhiều ứng dụng khác, với sự phát triển của các công nghệ như Node.js, React, Vue.js.



### Ngôn ngữ lập trình TypeScript

TypeScript là một ngôn ngữ lập trình mã nguồn mở được phát triển bởi Microsoft, là một phiên bản mở rộng của JavaScript. Nó giúp các lập trình viên viết mã JavaScript theo kiểu tĩnh (static typing), giúp phát hiện lỗi và cải thiện tính bảo trì của mã nguồn. TypeScript cung cấp các tính năng như các kiểu dữ liệu tĩnh, lớp, module và các tính năng nâng cao hơn so với JavaScript thuần. TypeScript cho phép phát triển ứng dụng lớn và phức tạp hơn một cách dễ dàng và an toàn hơn, đặc biệt là trong các dự án lớn và có nhiều thành viên tham gia. Các framework như Angular của Google và NestJS cũng sử dụng TypeScript làm ngôn ngữ chính để phát triển ứng dụng web và server-side.



## Những framework hổ trợ phát triển ứng dụng

### Framework Spring boot

Spring Boot là khuôn mẫu (framework) phát triển dựa trên ngôn ngữ Java trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên đơn giản hóa quá trình lập trình một ứng dụng với Spring. Trong Spring Boot, người lập trình chỉ tập trung vào việc phát triển các nghiệp vụ cho ứng dụng mà không quan tâm nhiều về kiến trúc và mã nguồn của ứng dụng. Một kiến trúc webserver xây dựng dựa trên spring boot gồm các thành phần cơ bản như sau: Repository Class Extending, Service Layer, Model, Controller.

### Framework ExpressJS

### Framework NestJS

### Framework NextJS

## Những công nghệ áp dụng

### Node.js

Node.js là một nền tảng phát triển ứng dụng được xây dựng dựa trên JavaScript, được sử dụng để xây dựng các ứng dụng mạng có khả năng mở rộng. Node.js sử dụng mô hình I/O không đồng bộ, cho phép xử lý nhiều yêu cầu mạng một cách hiệu quả. Trong dự án này, Node.js sẽ được sử dụng cho cả backend của ứng dụng di động và web, cho phép chúng ta xây dựng và quản lý các API, xử lý logic ứng dụng, và tương tác với cơ sở dữ liệu.



Hình 2‑1 Node.js development

### MongoDB / DynamoDB

MongoDB và DynamoDB đều là các hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) được sử dụng trong dự án này.

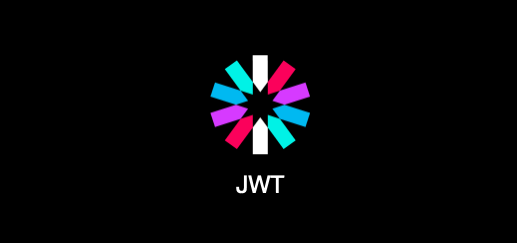
* MongoDB là một cơ sở dữ liệu mã nguồn mở, linh hoạt và dễ sử dụng, được thiết kế để lưu trữ dữ liệu dưới dạng tài liệu JSON có cấu trúc.
* DynamoDB là một dịch vụ cơ sở dữ liệu NoSQL do AWS cung cấp, nhanh chóng, linh hoạt và có khả năng mở rộng. Nó cung cấp các tính năng như tự động mở rộng, bảo mật tích hợp và hiệu suất cao.

Sự kết hợp giữa Node.js và MongoDB/DynamoDB sẽ cung cấp một nền tảng mạnh mẽ cho việc phát triển và quản lý dữ liệu trong ứng dụng chat của chúng ta trên cả hai nền tảng di động và web.



### JWT (Json Token Web)

JWT (JSON Web Token) là một tiêu chuẩn mở được sử dụng để tạo và truyền thông tin xác thực an toàn giữa các bên dưới dạng một đoạn mã JSON. JWT thường được sử dụng trong các ứng dụng web và di động để xác thực người dùng và quản lý phiên làm việc.

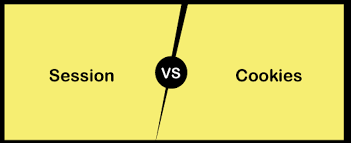


### Mailer Service

Dịch vụ mailer là các dịch vụ được cung cấp bởi các nhà cung cấp để hỗ trợ việc gửi và nhận email từ các ứng dụng và hệ thống. Các dịch vụ này thường được sử dụng bởi các doanh nghiệp và nhà phát triển để tạo ra và quản lý các chiến dịch gửi email, cũng như để xây dựng các hệ thống email tự động.

### Session/Cookies

Dùng để lưu trạng thái và tương tác, lưu trữ thông tin đăng nhập của người dùng.



# : 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 | Đây là những người dùng chính sử dụng website và app health-care để chăm sóc sức khỏe của họ.  Chức năng:   * Đăng nhập vào ứng dụng. * Đăng ký tài khoản. * Cập nhật thông tin cá nhân * Nhập thông tin sức khỏe (nhịp tim và huyết áp) vào hệ thống. * Xem lịch sử sức khỏe của bản thân. * Đặt lịch khám online hoặc ofline với bác sĩ. * Tham gia vào các phòng meeting để gặp bác sĩ hoặc chuyên gia tư vấn. * Thanh toán * Giao tiếp với AI về các vấn đề sức khỏe. * Tìm kiếm và xem các cẩm nang về sức khỏe * Tham gia các forum để đặt câu hỏi và chia sẻ về chăm sóc sức khỏe |
| Doctor | Đây là những người dùng sử dụng website và app health-care để tự vấn cung cấp dịch vụ chăm sóc sức khỏe chất lượng cao và tương tác trực tiếp với bệnh nhân để cải thiện chất lượng dịch vụ y tế.  Chức năng:   * Đăng nhập vào ứng dụng. * Cập nhật thông tin cá nhân. * Thu thập thông tin người dùng. * Đặt lịch hẹn với bệnh nhân. * Quản lý các lịch hẹn . * Xem lịch sử sức khỏe của người bệnh. * Tham gia vào các phòng meeting để các bệnh nhân. * Tạo các hồ sơ bệnh án. * Giao tiếp với AI về các vấn đề sức khỏe. * Tạo ra các sổ tay y tế để người bệnh tham khảo. * Tham gia các forum để giải đáp thắc mắc cho người bệnh. |
| Admin |  |

Bảng 3‑1 Danh sách tác nhân và mô tả

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

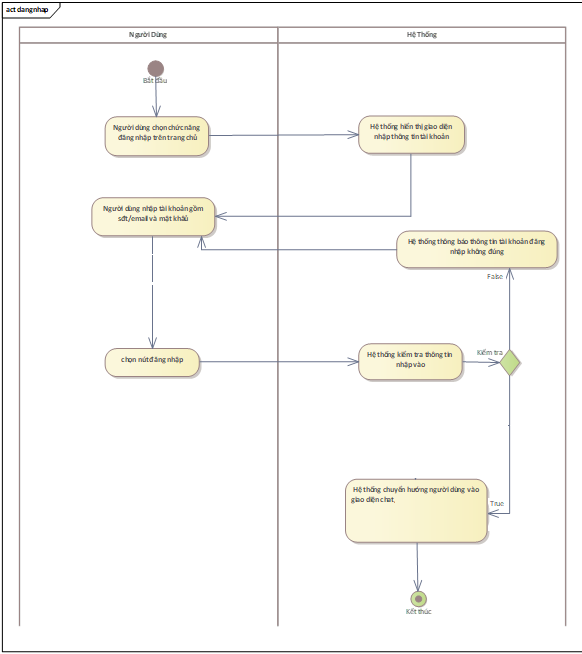
|  |  |
| --- | --- |
| ID | Tên Use case |
| UC001 | Đăng nhập vào app/website health-care |
| UC002 | Đăng ký |
| UC003 |  |
| UC004 |  |
| UC005 |  |
| UC006 |  |
| UC007 |  |
| UC008 |  |
| UC009 |  |
| UC010 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

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

### Tình huống hoạt động

#### UC001: Đăng nhập vào app/website health-care

|  |  |
| --- | --- |
| **Tên use case**: Đăng nhập vào app/website health-care | |
| **Mô tả sơ lược**: Người dùng muốn đăng nhập vào hệ thống để sử dụng các dịch vụ | |
| **Actor chính**: Người dùng | |
| **Actor phụ**: | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| **Hậu điều kiện (Post-condition):** | |
| **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn chức năng đăng nhập trên trang chủ | 1. Hệ thống hiển thị giao diện nhập thông tin tài khoản |
| 1. Người dùng nhập tài khoản gồm sđt/email và mật khẩu |  |
| 1. Chọn nút đăng nhập | 1. Hệ thống kiểm tra thông tin nhập vào |
|  | 1. Hệ thống chuyển hướng người dùng vào giao diện |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 5.1. Hệ thống thông báo thông tin tài khoản đăng nhập không đúng |
|  | 5.2. Hệ thống quay lại bước 3 |



A diagram of a server

Description automatically generated

* Usecase đăng ký

|  |  |
| --- | --- |
| * **Tên use case**: UC02\_Đăng ký | |
| * **Mô tả sơ lược**: Người dùng muốn đăng ký vào hệ thống để sử dụng các dịch vụ | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng chưa có tài khoản | |
| * **Hậu điều kiện (Post-condition):** người dùng tạo tài khoản thành công | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn chức năng đăng ký trên trang chủ | 1. Hiển thị giao diện đăng ký |
| 1. Người dùng nhập thông tin gồm tên, email/sđt, mật khẩu |  |
| 1. Người dùng nhấn đăng ký | 1. Hệ thống kiểm tra thông tin nhập vào |
|  | 1. Hệ thống kiểm tra có tồn tại tài khoản chưa |
|  | 1. Hệ thống gửi mã xác nhận |
| 1. Người dùng nhập mã OTP |  |
| 1. Người dùng bấm xác nhận | 1. Hệ thống kiểm tra OTP có hợp lệ không |
|  | 1. Thông báo đang ký thành công và chuyển hướng tới giao diện chat |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 5.1. Hệ thống thông báo nhập không đúng định dạng |
|  | 5.2. Hệ thống quay lại bước 3 |
|  | 6.1. Hệ thống thông báo đã tồn tại |
|  | 6.2 Hệ thống quay lại bước 3 |
|  | 10.1 hệ thống thông báo OTP không hợp lệ |
|  | 10.2 hệ thống quay lại bước 8 |



A diagram of a diagram

Description automatically generated

* Use case quên mật khẩu

|  |  |
| --- | --- |
| * **Tên use case**: UC03\_ Quên mật khẩu | |
| * **Mô tả sơ lược**: Người dùng lấy lại mật khẩu đã mất | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** Người dùng lấy lại được mật khẩu mới | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn chức năng quên mật khẩu. | 1. Hiển thị giao diện lấy lại mật khẩu |
| 1. Người dùng nhập email và bấm xác nhận | 1. Hệ thống kiểm tra tài khoản đã tồn tại chưa |
|  | 1. Hệ thống hiển thị giao diện nhập mật khẩu mới và gửi mã OTP |
| 1. Người dùng nhập mật khẩu mới và nhập mã OTP | 1. Hệ thống kiểm tra OTP và mật khẩu mới đã nhập |
|  | 1. Hệ thống xác nhận đổi mật khẩu thành công và quay về trang đăng nhập. |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4.1. Hệ thống thông báo không tồn tại |
|  | 4.2. Hệ thống quay lại bước 3 |
|  | * 1. Hệ thống thông báo mật khẩu không hợp lệ hoặc mã OTP sai |
|  | 7.2 Hệ thống quay lại bước 6 |

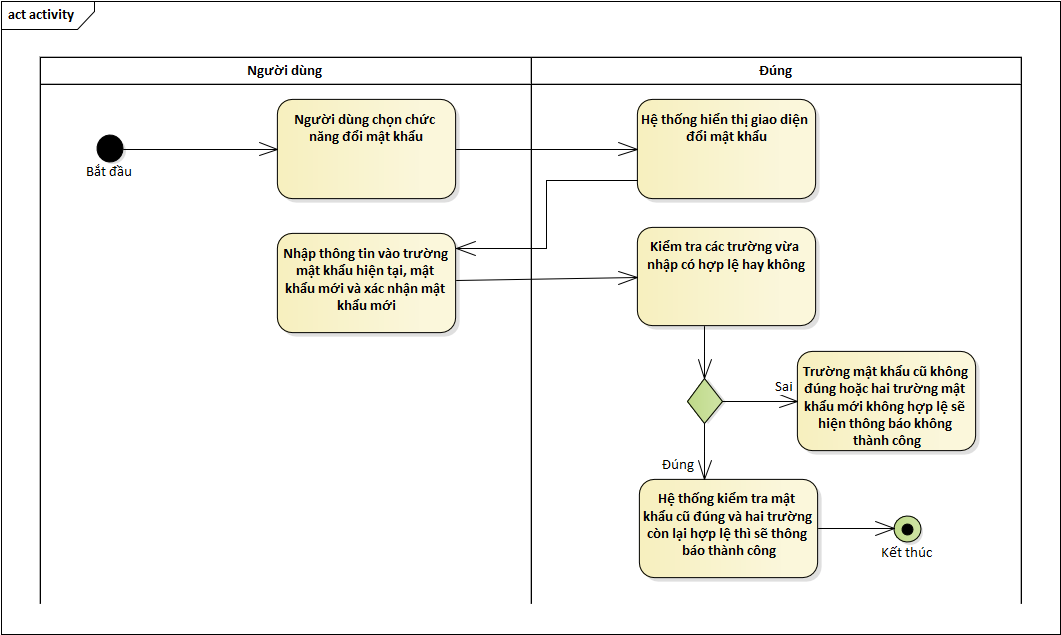
* Usecase cập nhật thông tin người dùng

|  |  |
| --- | --- |
| * **Tên use case**: UC04\_Cập Nhập Thông Tin Người Dùng | |
| * **Mô tả sơ lược**: Giúp người cập nhập thông tin của người dùng trong hệ thông | |
| * **Actor chính**: người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đăng nhập thành công vào hệ thống | |
| * **Hậu điều kiện (Post-condition):** Thông tin người dùng được cập nhập | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn chức năng cập nhật thông tin | 1. Hệ thống hiển thị giao diện quản lí thông tin người dùng |
| 1. Người dùng nhập thông tin tài khoản cần thay đổi (avatar nếu có) |  |
| 1. Người dùng nhấn cập nhật | 1. Hệ thống kiểm tra thông tin nhập vào |
|  | 1. Thông báo cập nhập thành công |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 5.1. Hệ thống thông báo thông tin không hợp lệ |
|  | 5.2. Hệ thống quay lại bước 3 |



* Usecase đổi mật khẩu

|  |  |
| --- | --- |
| * **Tên use case:** UC05\_Đổi mật khẩu | |
| * **Mô tả sơ lược:** Đổi mật khẩu cho tài khoản cá nhân | |
| * **Actor chính:** Người dùng | |
| * **Actor phụ: Không** | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống | |
| * **Hậu điều kiện (Post-condition):** Mật khẩu của người dùng được thay đổi và lưu lại trong hệ thống. | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng nhấn nút đổi mật khẩu. | 1. Hệ thống hiển thị giao diện đổi mật khẩu |
| 1. Nhập thông tin vào trường mật khẩu hiện tại, mật khẩu mới và xác nhận mật khẩu mới | 1. Kiểm tra các trường vừa nhập có hợp lệ hay không |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4.1 Trường mật khẩu cũ không đúng hoặc hai trường mật khẩu mới không hợp lệ sẽ hiện thông báo không thành công |
|  | 4.2 Hệ thống kiểm tra trường mật khẩu cũ đúng và hai trường còn lại hợp lệ thì sẽ thông báo thành công |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |



* Usecase tìm người dùng

|  |  |
| --- | --- |
| **Use case:** UC06\_Tìm người dùng | |
| **Mô tả sơ lược**: Giúp người sử dụng tìm một người dùng khác thông qua số điện thoại | |
| **Actor chính**: Người dùng. | |
| **Actor phụ**: Không. | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống và đang ở trang chủ. | |
| **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn vào biểu tượng “thêm bạn” ở màn hình chính | 1. Hệ thống hiển thị Modal tìm kiếm |
| 1. Người dùng nhập số điện thoại của người dùng mà mình muốn tìm kiếm |  |
| 1. Người dùng bấm nút tìm kiếm | 1. Hệ thống kiểm tra tính hợp lệ của số điện thoại |
|  | 1. Hệ thống gửi yêu cầu tìm kiếm đến server |
|  | 1. Hệ thống nhận kết quả trả về và hiển thị thông tin người dùng tìm được lên Modal |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 5.1 Hệ thống thông báo số điện thoại không hợp lệ |
|  | 5.2 Quay lại bước 3 |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |



* Usecase tìm kiếm cuộc trò chuyện

|  |  |
| --- | --- |
| **Use case:** UC07\_Tìm kiếm trò chuyện | |
| **Mô tả sơ lược**: Giúp người dùng tìm kiếm các cuộc trò chuyện theo tên | |
| **Actor chính**: Người dùng. | |
| **Actor phụ**: Không. | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống và đang ở trang chủ. | |
| **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng nhập tên cuộc trò truyện muốn tìm kiếm vào thanh tìm kiếm ở giao diện trang chủ | 1. Hệ thống gửi yêu cầu đến server |
|  | 1. Hệ thống nhận phản hồi từ server và hiển thị danh sách cuộc trò truyện tìm kiếm được trên thanh danh sách các cuộc trò truyện |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 3.1 Hệ thống hiển thị thông báo không tìm thấy cuộc trò truyện phù hợp |
|  | 3.2 Quay lại bước 1 |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |



* Usecase tạo cuộc trò chuyện nhóm

|  |  |
| --- | --- |
| **Use case:** UC08\_Tạo cuộc trò truyện nhóm | |
| **Mô tả sơ lược**: Giúp người dùng tạo một cuộc trò chuyện với nhiều thành viên | |
| **Actor chính**: Người dùng. | |
| **Actor phụ**: Không. | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống và đang ở trang chủ. | |
| **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn vào biểu tượng tạo nhóm ở trang chủ | 1. Hệ thống hiển thị Modal tạo nhóm |
| 1. Người dùng chọn các thành viên để thêm vào nhóm |  |
| 1. Người dùng nhập tên nhóm | 1. Hệ thống kiểm tra tính hợp lệ của tên nhóm(không bỏ trống) |
| 1. Người dùng bấm vào nút “tạo nhóm” | 1. Hệ thống gửi yêu cầu đến server |
|  | 1. Hệ thống tạo nhóm mới và thêm nhóm vừa tạo vào danh sách nhóm. |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 5.1 Hệ thống thông báo tên nhập không hợp lệ |
|  | 5.2 Hệ thống quay lại bước 4 |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |



* Usecase nhắn tin.

|  |  |
| --- | --- |
| **Use case:** UC09\_Nhắn tin | |
| **Mô tả sơ lược**: Nhắn tin với bạn bè hoặc trong một nhóm chat | |
| **Actor chính**: Người dùng | |
| **Actor phụ**: Không. | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống và đã tạo cuộc trò chuyện với bạn bè hoặc nhóm muốn nhắn tin. | |
| **Hậu điều kiện (Post-condition):** Người dùng có thể nhắn tin với bạn bè hoặc nhóm trong cửa sổ chat. | |
| * **Luồng sự kiện chính (main flow):** | |
| **Người dùng** | **Hệ thống** |
| 1. Người dùng chọn bạn bè hoặc nhóm muốn gửi tin nhắn | 1. Hệ thống hiển thị giao diện nhắn tin và danh sách tin nhắn |
| 1. Người dùng nhập tin nhắn muốn gửi và nhấn nút gửi | 1. Hệ thống kiểm tra tin nhắn và xử lý gửi tin nhắn |
|  | 1. Hệ thống xóa nội dung tin nhắn trong ô nhập tin nhắn |
|  | 1. Hệ thống cập nhật lại danh sách tin nhắn và hiển thị nội dung tin nhắn lên giao diện nhắn tin |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4.1. Hệ thống kiểm tra bội dung tin nhắn không hợp lệ (rỗng, chỉ chứa kí tự khoảng trắng) |
|  | 4.2. Hệ thống xóa nội dung tin nhắn trong ô nhập tin nhắn |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |
|  |  |



* Usecase Gửi yêu cầu kết bạn

|  |  |
| --- | --- |
| **Use case:** UC10\_Gửi yêu cầu kết bạn | |
| **Mô tả sơ lược**: Người dùng gửi yêu cầu kết bạn | |
| **Actor chính**: Người dùng | |
| **Actor phụ**: Không. | |
| **Tiền điều kiện (Pre-condition):** Người dùng đã đăng nhập vào hệ thống và đang ở giao diện bạn bè; | |
| **Hậu điều kiện (Post-condition):** Người dùng có thể gửi yêu cầu kết bạn cho bạn bè | |
| * **Luồng sự kiện chính (main flow):** | |
| **Người dùng** | **Hệ thống** |
| 1. Người dùng chọn chức năng thêm bạn bè | 1. Hệ thống hiển thị giao diện tìm kiếm |
| 1. Người dùng nhập số điện thoại 2. của người cần kết bạn và nhấn nút tìm kiếm | 1. Hệ thống xử lý tìm kiếm |
|  | 1. Hệ thống hiển thị thông tin của người cần kết bạn |
| 1. Người dùng chọn nút kết bạn | 1. Hệ thống xử lý gửi yêu cầu kết bạn |
|  | 1. Hệ thống thông báo “gửi yêu cầu kết bạn thành công” |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4.1 Hệ thống thông báo “không tìm thấy” |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |
|  |  |

****

Activity diagram.

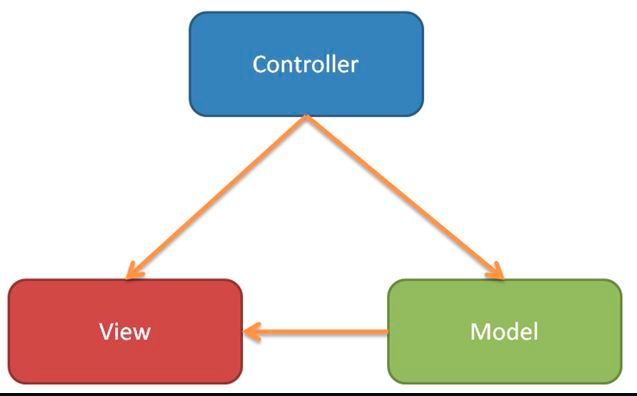
Sequence diagram cho từng tình huống.

## Class diagram

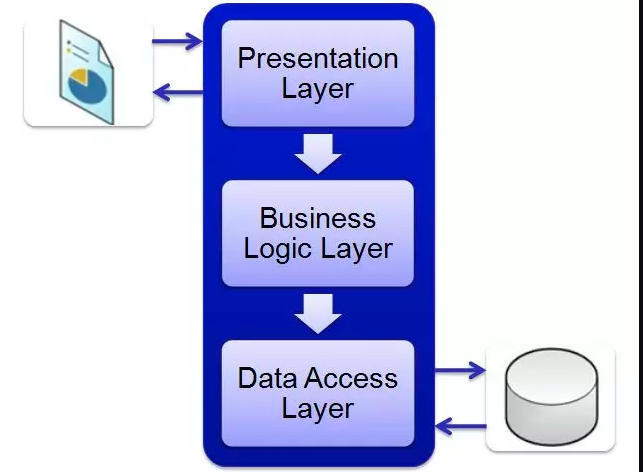
Mô hình lớp (Mô hình lớp đầy đủ + Mô hình lớp bao gồm các lớp Entity)

## Deployment diagram

Áp dụng kiến trúc Microservice kết hợp với sử dụng các service để tách biệt logic các ứng dụng ra khỏi lớp controller. Đây là một kỹ thuật thường thấy trong kiến trúc phần mềm để tăng cường tính mô-đun và dễ dàng bảo trì hơn.



Kiến trúc lớp ba tầng (Three layers):

* Tầng trình bày(Presentation layer): Được đại diện bởi Controller, xử lý các yêu cầu từ người dùng và trả về kết quả.
* Tầng nghiệp vụ(Business Logic Layer) : Bao gồm các lớp như Iservice và Service, nơi các logic xử lý chính của ứng dụng được thực hiện.
* Tầng dữ liệu (Data Access Layer): Được thể hiện qua Model, nơi xử lý trực tiếp các hoạt động liên quan đến cơ sở dữ liệu.
* 

**Cấu trúc file code**

- Controller: Xử lý các yêu cầu đến từ người dùng, sau đó gọi đến các services để xử lý logic nghiệp vụ. Controller sẽ làm nhiệm vụ điều hướng và trung gian giữa người dùng và dữ liệu ứng dụng.

* IService: Là interface (giao diện) định nghĩa các phương thức mà service liên quan phải cung cấp. Sử dụng interface để giúp tăng tính trừu tượng và làm cho việc kiểm thử dễ dàng hơn.
* Service: Là lớp cụ thể thực hiện các phương thức đã được định nghĩa trong Iservice. Lớp này sẽ xử lý các nghiệp vụ chính liên quan đến “phòng”, như quản lý thông tin, tìm kiếm, cập nhật, và xoá dữ liệu về phòng.
* DTO (Data Transfer Object): Đây là một đối tượng mà qua đó dữ liệu được truyền giữa csac lớp và các tầng trong ứng dụng. DTO thường không chứa bất kỳ logic nghiệp vụ nào mà chỉ mang dữ liệu.
* Model: Chứa các định nghĩa của dữ liệu và thường là những đối tượng mà ánh xạ trực tiếp tới cấu trúc bảng trong cơ sở dữ liệu. Model có thể chứa một số logic liên quan đến validator dữ liệu.

# : HIỆN THỰC

## Cấu hình phần cứng, phần mềm

* Dịch vụ lưu trữ hình ảnh Cloudinary
* Cấu hình Server: Visual code studio,IntelJ, MongoDB Atlas, MySql
* Cấu hình Client: Visual code studio

## Giao diện của hệ thống

## Kế hoạch và hiện thực kiểm thử hệ thống

### Kế hoạch kiểm thử

#### Phần cứng: Máy tính cá nhân có kết nối mạng LAN

|  |  |  |  |
| --- | --- | --- | --- |
| **CPU** | **RAM** | **HDD** | **Architecture** |
| Intel Core i5, 1.3 GHz trở lên | 4 GB | 120 GB | 64bit |

Bảng 4- 1 phần cứng

#### Phần mềm

|  |  |  |
| --- | --- | --- |
| **Tên phần mềm** | **Phiên bản** | **Loại** |
| Visual studio code | 1.62.0 | IDE cho nodejs & reactjs |
| MongoDB | 5.0.2 2008R2 Plus | Hệ quản trị cơ sở dữ liệu |
| Microsoft Windows 10 | 10 | Hệ điều hành |

### Kiểm thử hệ thống

https://docs.google.com/spreadsheets/d/1A\_CJ8sOcwsikauyn6hOqEePx2FFHlI1RUPFkPJFld24/edit?usp=sharing

# : KẾT LUẬN

## Kết quả đạt được

## Hạn chế của đồ án

## Nhóm chưa làm xong phần admin, video call

## Hướng phát triển

# 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>