

TRƯỜNG ĐẠI HỌC KINH TẾ  
**KHOA THỐNG KÊ – TIN HỌC**



**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**  
**NGÀNH HỆ THỐNG THÔNG TIN QUẢN LÝ**  
**CHUYÊN NGÀNH TIN HỌC QUẢN LÝ**

**Tham gia xây dựng thông báo cho hệ thống Quản trị dự án Zino sử dụng private Framework HNV-NewLine và SpringBoot**

Sinh viên thực hiện : **Phạm Thị Thương**

Lớp : **47k14**

Đơn vị thực tập : **Công ty cổ phần INOTEV**

Cán bộ hướng dẫn : **Hoàng Nguyên Vũ**

Giảng viên hướng dẫn : **TS. Đặng Trung Thành**

***Đà Nẵng, 8/2024***

## NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP

### NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP

Họ và tên sinh viên: Phạm Thị Thương

Lớp: 47K14

Khoa Thống kê – Tin học, Trường Đại học Kinh tế, Đại học Đà Nẵng

Thực tập từ ngày: 02 /05 /2024 đến ngày: 02 /08 /2024

Tên đơn vị thực tập: Công ty cổ phần INOTEV

Địa chỉ: 62 Hồ Quý Ly, Phường Thanh Khê Tây, Quận Thanh Khê, Đà Nẵng

Số điện thoại liên hệ: 0845204946

Họ tên cán bộ hướng dẫn: Hoàng Nguyên Vũ

Sau quá trình thực tập của sinh viên tại đơn vị, chúng tôi có một số đánh giá như sau:

STT	Nội dung đánh giá	Rất không tốt	Không tốt	Bình thường	Tốt	Rất tốt
1	Về thái độ, ý thức, đạo đức và việc tuân thủ các quy định và văn hóa đơn vị thực tập	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	X
2	Kiến thức chuyên môn	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	X	<input type="checkbox"/>
3	Khả năng hòa nhập, thích nghi và tác phong nghề nghiệp	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	X
4	Trách nhiệm, sáng tạo trong công việc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	X	<input type="checkbox"/>

(Anh/chị vui lòng đánh dấu X vào ô tương ứng với năng lực của sinh viên)

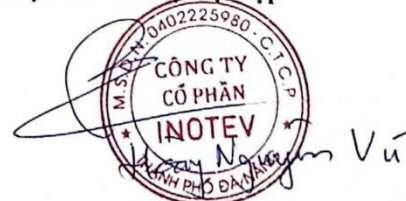
**Ý kiến nhận xét và đề xuất** (Nhằm nâng cao chất lượng đào tạo, Nhà trường rất mong muốn nhận thêm những ý kiến khác từ quý doanh nghiệp):

Thái độ tốt, tuân thủ đúng các quy định của đơn vị, tham gia đầy đủ các hoạt động của công ty .....

.....  
.....  
.....  
.....

Đà Nẵng, ngày 10 tháng 07 năm 2024

Xác nhận của đơn vị thực tập



Được quét bằng CamScanner

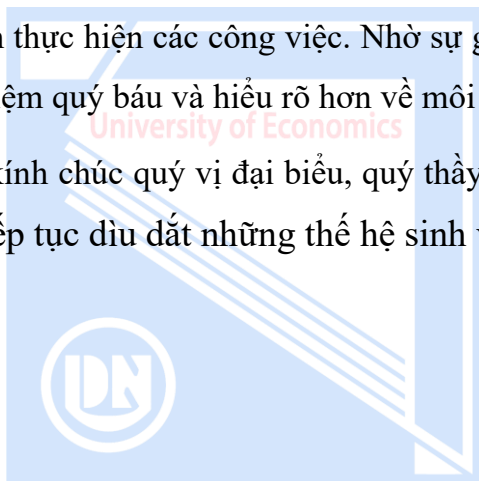
## LỜI CẢM ƠN

Lời đầu tiên, cho phép em được gửi lời tri ân sâu sắc đến Ban Giám đốc, quý thầy cô trong Khoa Thống kê – Tin học giúp đỡ em thực hiện báo cáo thực tập kì hè này.

Thành công của bài báo cáo thực tập này không thể thiếu sự đóng góp to lớn của thầy TS. Đặng Trung Thành - Khoa Thống kê – Tin học, trường Đại học Kinh tế - Đại học Đà Nẵng. Em xin trân trọng cảm ơn thầy đã tận tình hướng dẫn, chỉ bảo và luôn dành cho em những lời khuyên bổ ích trong suốt quá trình thực hiện báo cáo. Nhờ sự dìu dắt ân cần của thầy, em đã có thể hoàn thành bài báo cáo một cách hiệu quả và đầy đủ.

Em cũng xin gửi lời cảm ơn chân thành đến quý công ty cổ phần INOTEV đã tạo điều kiện cho em thực tập tại công ty, giúp em có cơ hội cọ xát thực tế và trau dồi kiến thức chuyên môn. Em đặc biệt biết ơn thầy Hoàng Nguyên Vũ đã luôn nhiệt tình giải đáp thắc mắc và hướng dẫn em cách thực hiện các công việc. Nhờ sự giúp đỡ của quý công ty, em đã có thêm nhiều kinh nghiệm quý báu và hiểu rõ hơn về môi trường làm việc thực tế.

Cuối cùng, em xin kính chúc quý vị đại biểu, quý thầy cô sức khỏe dồi dào, thành công trong công việc và tiếp tục dìu dắt những thế hệ sinh viên sau này.



## LỜI CAM ĐOAN

Em xin cam đoan đề tài “Xây dựng chức năng trò chuyện cho hệ thống Zino” là một đề tài được lấy từ dự án thực tế của Zino mà em đang góp phần tham gia dưới sự cho phép của thầy Hoàng Nguyên Vũ – Giám đốc của công ty INOTEV. Ngoài ra, không có bất cứ sự sao chép của người khác. Đề tài, nội dung báo cáo thực tập là sản phẩm mà em đã nỗ lực nghiên cứu trong quá trình học tập tại trường cũng như tham gia thực tập tại Công ty INOTEV. Các số liệu, kết quả trình bày trong báo cáo là hoàn toàn trung thực, em xin chịu hoàn toàn trách nhiệm, kỷ luật của bộ môn và nhà trường đề ra nếu như có vấn đề xảy ra.



## MỤC LỤC

<b>NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP .....</b>	<b>i</b>
<b>LỜI CẢM ƠN .....</b>	<b>ii</b>
<b>LỜI CAM ĐOAN .....</b>	<b>iii</b>
<b>MỤC LỤC .....</b>	<b>iv</b>
<b>DANH MỤC HÌNH ẢNH .....</b>	<b>vii</b>
<b>DANH MỤC BẢNG BIỂU .....</b>	<b>ix</b>
<b>DANH MỤC CÁC TỪ VIẾT TẮT .....</b>	<b>x</b>
<b>LỜI MỞ ĐẦU .....</b>	<b>1</b>
<b>CHƯƠNG 1. TỔNG QUAN VỀ CÔNG TY .....</b>	<b>3</b>
1.1. Giới thiệu tổng quát về doanh nghiệp thực tập .....	3
1.1.1 Thông tin chung .....	3
1.1.2 Dịch vụ .....	3
1.2. Tổng quan về vị trí việc làm .....	4
1.2.1 Thời gian làm việc.....	4
1.2.2 Tổng quan về vị trí thực tập Developer Full Stack Java.....	4
<b>CHƯƠNG 2. TỔNG QUAN VỀ LÝ THUYẾT .....</b>	<b>6</b>
2.1. Giới thiệu về private Framework HNV-NewLine và SpringBoot. ....	6
2.1.1 Private Framework HNV-NewLine .....	6
2.1.2 SpringBoot. [6].....	7
2.2. Tìm hiểu về các kiến trúc phần mềm phù hợp cho ứng dụng quản lý dự án. ....	8
2.2.1 Kiến trúc nhiều lớp (Multi-tier architecture). [7] .....	8
2.2.2 Kiến trúc vi mô dịch vụ (Microservices architecture) [8] .....	9
2.2.3 Kiến trúc Hexagonal (Hexagonal architecture). [9] .....	9
2.3. Các mô hình dữ liệu và quản lý dữ liệu trong ứng dụng quản lý dự án.....	10

2.3.1	Các mô hình dữ liệu trong Zino.....	10
2.3.2	Quản lý dữ liệu trong Zino.....	10
2.4.	Các phương pháp bảo mật cho ứng dụng web. ....	11
2.4.1	Xác thực và ủy quyền.....	11
2.4.2	Mã hóa dữ liệu.....	11
<b>CHƯƠNG 3.</b>	<b>GIỚI THIỆU DỰ ÁN VÀ PHẦN MỀM.....</b>	<b>12</b>
3.1.	Giới thiệu về dự án quản lý dự án Zino. ....	12
3.1.1	Tổng quan về dự án Zino. ....	12
3.1.2	Lợi ích của dự án Zino. ....	12
3.1.3	Chức năng chính của dự án. ....	13
3.2.	Giới thiệu Phần mềm Eclipse và ngôn ngữ lập trình Java, cùng các công cụ khác. ....	14
3.2.1	Giới thiệu Phần mềm Eclipse. [10] .....	14
3.2.2	Ngôn ngữ lập trình Java. [11] .....	15
3.2.3	My SQL. [12] .....	16
<b>CHƯƠNG 4.</b>	<b>PHÂN TÍCH THIẾT KẾ HỆ THỐNG .....</b>	<b>18</b>
4.1.	Phân tích yêu cầu hệ thống.....	18
4.1.1	Đặc tả các trường hợp yêu cầu thông báo. ....	18
4.1.2	Use case.....	19
4.1.3	Đặc tả use case .....	19
4.2.	Thiết kế cơ sở dữ liệu.....	26
4.2.1	Mức khái niệm .....	26
4.2.2	Mức logic .....	27
4.2.3	Mức vật lý .....	28
4.3.	Thiết kế giao diện và đặc tả giao diện.....	28
4.3.1	Thiết kế giao diện.....	29
4.3.2	Đặc tả giao diện.....	29



## DANH MỤC HÌNH ẢNH

Hình 1.1 Logo công ty cổ phần INOTEV .....	3
Hình 2.1. Đặc điểm của Sprint Boot.....	8
Hình 2.2 Kiến trúc ứng dụng web zino sử dụng.....	8
Hình 3.1. Eclipse. ....	15
Hình 3.2. Java .....	15
Hình 4.1. Use case quản lý thông báo .....	19
Hình 4.2. Luồng nhận thông báo. ....	21
Hình 4.3. Luồng xem thông báo .....	23
Hình 4.4. Luồng xóa thông báo. ....	25
Hình 4.5. Luồng đánh dấu đã xem .....	26
Hình 4.6. CSDL mức vật lý.....	28
Hình 4.7: Giao diện thông báo. ....	29
Hình 4.8. Đặc tả giao diện thông báo .....	29
Hình 5.1. Hình Cấu trúc Framework .....	31
Hình 5.2. Hình Cấu trúc của controller .....	31
Hình 5.3. Cách khởi tạo package.....	32
Hình 5.4. Sau khi khởi tạo .....	32
Hình 5.5. Dữ liệu json nhận được của thông báo thêm dự án/ phân việc. ....	34
Hình 5.6. Dữ liệu json nhận được của thông báo thêm comment .....	35
Hình 5.7 Dữ liệu json nhận được của thông báo sửa thông tin dự án .....	36
Hình 5.8. Dữ liệu json nhận được của thông báo chuyển phân việc .....	37
Hình 5.9. Dữ liệu json nhận được của thông báo thêm thành viên. ....	38
Hình 5.10. Dữ liệu json nhận được của thông báo thêm mới lịch hẹn.....	39
Hình 5.11. Dữ liệu json nhận được của thông báo thêm mới tin tức. ....	40
Hình 5.12. Dữ liệu json nhận được của thông báo bài viết được duyệt. ....	40
Hình 5.13 Dữ liệu json nhận được của thông báo phân việc trễ hẹn. ....	42
Hình 5.14. Dữ liệu json nhận được của thông báo phân việc sắp trễ hẹn. ....	43



Hình 5.15. Dữ liệu json nhận được của thông báo sắp trễ hẹn.....	44
Hình 5.16. Dữ liệu json nhận được của thông báo lịch hẹn sắp diễn ra.....	46
Hình 5.17. Code kết nối BE với CSDL .....	46
Hình 5.18. Code khởi chạy localhoots. ....	47



## DANH MỤC BẢNG BIỂU

Bảng 4.1. Đặc tả chi tiết use case nhận thông báo. ....	21
Bảng 4.2. Đặc tả chi tiết use case xem thông báo. ....	22
Bảng 4.3. Đặc tả chi tiết use case xóa thông báo. ....	24
Bảng 4.4. Đặc tả chi tiết use case đánh dấu đã xem thông báo. ....	26
Bảng 4.5. CSDL mức khái niệm .....	27
Bảng 4.6. CSDL mức logic .....	28
Bảng 4.8: Đặc tả giao diện thông báo.....	30
Bảng 5.1. API sử dụng trong chức năng thông báo.....	33



## DANH MỤC CÁC TỪ VIẾT TẮT

STT	Ký hiệu chữ viết tắt	Chữ viết đầy đủ
1	CSDL	Cơ sở dữ liệu
2	API	Application Programming Interface
3	FE	Font-end
4	BE	Back-end
5	CNTT	Công nghệ thông tin
6	CRM	Customer Relationship Management



# LỜI MỞ ĐẦU

## 1. Mục tiêu của đề tài

Nghiên cứu và phát triển thành công ứng dụng quản lý dự án sử dụng private framework HNV-NewLine và SpringBoot.

Đánh giá hiệu quả và tính ứng dụng của phần mềm trong thực tế.

## 2. Đối tượng và phạm vi nghiên cứu

### 2.1. Đối tượng nghiên cứu:

+ Doanh nghiệp: Zino có thể được sử dụng bởi các doanh nghiệp thuộc mọi quy mô và ngành nghề để quản lý các dự án của họ.

+ Tổ chức phi lợi nhuận: Zino có thể được sử dụng bởi các tổ chức phi lợi nhuận để quản lý các dự án cộng đồng, dự án thiện nguyện,...

+ Cá nhân: Zino có thể được sử dụng bởi cá nhân để quản lý các dự án cá nhân, dự án học tập,...

### 2.2. Phạm vi nghiên cứu:

+ Phân tích yêu cầu của người dùng.

+ Thiết kế hệ thống Zino.

+ Phát triển hệ thống Zino sử dụng private Framework HNV-NewLine và SpringBoot.

+ Thử nghiệm và hoàn thiện hệ thống Zino.

+ Tài liệu hướng dẫn sử dụng hệ thống Zino.

### 2.3. Thời gian và địa điểm.

- Thời gian thực hiện: 02/05/2024 đến 04/08/2024.

- Địa điểm: 62 Hồ Quý Ly, Phường Thanh Khê Tây, Quận Thanh Khê, Đà Nẵng.

## 3. Kết cấu của đề tài

Đề tài được tổ chức gồm phần mở đầu, 4 chương nội dung và phần kết luận:

- + **Mở đầu.**
- + **Chương 1:** Tổng quan về công ty.
- + **Chương 2:** Cơ sở lý thuyết.
- + **Chương 3:** Giới thiệu phần mềm và Framework.
- + **Chương 4:** Phân tích thiết kế hệ thống.
- + **Chương 5:** Phát triển ứng dụng với chức năng thông báo.
- + **Kết luận và hướng phát triển.**



## CHƯƠNG 1. TỔNG QUAN VỀ CÔNG TY

### 1.1. Giới thiệu tổng quát về doanh nghiệp thực tập

#### 1.1.1 Thông tin chung

INOTEV là một công ty cổ phần có trụ sở chính tại Pháp, chuyên cung cấp các giải pháp công nghệ thông tin (CNTT) cho doanh nghiệp. Công ty được thành lập vào năm 2002 với đội ngũ nhân viên giàu kinh nghiệm và chuyên môn cao. INOTEV cam kết cung cấp cho khách hàng các giải pháp CNTT tiên tiến, hiệu quả và tiết kiệm chi phí để giúp họ đạt được mục tiêu kinh doanh [1].



Hình 1.1 Logo công ty cổ phần INOTEV

Logo này được thiết kế để thể hiện những giá trị cốt lõi của công ty INOTEV, bao gồm:

- Sự đổi mới: INOTEV luôn cam kết cung cấp cho khách hàng các giải pháp CNTT tiên tiến nhất.
- Chất lượng: INOTEV cam kết cung cấp cho khách hàng dịch vụ chất lượng cao và hỗ trợ khách hàng chu đáo.
- Sự tin cậy: INOTEV cam kết là một đối tác đáng tin cậy cho khách hàng.
- Sự hợp tác: INOTEV cam kết hợp tác chặt chẽ với khách hàng để hiểu nhu cầu của họ và cung cấp cho họ các giải pháp phù hợp.

Logo của công ty INOTEV là một biểu tượng mạnh mẽ và dễ nhận biết, thể hiện sự cam kết của công ty đối với đổi mới, chất lượng, sự tin cậy và hợp tác. [2]

#### 1.1.2 Dịch vụ

- Bảo mật & Di động Doanh nghiệp.
- Microsoft 365.
- Quy trình Doanh nghiệp.
- Khách hàng (CRM).
- Tài chính.
- Nhân tài.
- Cloud.
- Hạ tầng.
- Phân tích Dữ liệu.

## **1.2. Tổng quan về vị trí việc làm**

### **1.2.1 Thời gian làm việc**

- Thứ 2: Buổi sáng: 8 giờ 30 - 12 giờ; Buổi chiều: 13 giờ 30 – 17 giờ 30
- Thứ 3: Buổi sáng: 8 giờ 30 - 12 giờ; Buổi chiều: 13 giờ 30 – 17 giờ 30
- Thứ 4: Buổi sáng: 8 giờ 30 - 12 giờ; Buổi chiều: 13 giờ 30 – 17 giờ 30
- Thứ 5: Buổi sáng: 8 giờ 30 - 12 giờ; Buổi chiều: 13 giờ 30 – 17 giờ 30
- Thứ 6: Buổi sáng: 8 giờ 30 - 12 giờ; Buổi chiều: 13 giờ 30 – 17 giờ 30
- Thứ 7: Nghỉ
- Chủ Nhật: Nghỉ

### **1.2.2 Tổng quan về vị trí thực tập Developer Full Stack Java.**

Developer Full Stack Java là lập trình viên có khả năng phát triển phần mềm toàn diện, bao gồm cả phần giao diện người dùng (front-end) và phần logic xử lý (back-end) bằng ngôn ngữ lập trình Java. Có kiến thức chuyên sâu về cả hai mảng công nghệ này và có thể làm việc độc lập để hoàn thiện một dự án phần mềm. [3]

#### ***a. Công việc chính của Developer Full Stack Java: [4]***

- **Phát triển front-end:**
  - + Thiết kế và xây dựng giao diện người dùng bằng HTML, CSS và JavaScript.
  - + Tạo các ứng dụng web tương tác và đáp ứng với các thiết bị khác nhau.

- + Viết mã JavaScript để xử lý các sự kiện người dùng và tương tác với back-end.

- **Phát triển back-end:**

- + Thiết kế và xây dựng API RESTful bằng Java.
- + Viết mã Java để xử lý logic nghiệp vụ của ứng dụng.
- + Kết nối với cơ sở dữ liệu để lưu trữ và truy xuất dữ liệu.
- + Triển khai ứng dụng web lên máy chủ.
- Bảo trì và nâng cấp ứng dụng:
  - + Sửa lỗi và cải thiện hiệu suất của ứng dụng.
  - + Cập nhật ứng dụng với các tính năng mới.
  - + Đảm bảo ứng dụng hoạt động ổn định và an toàn.

**b. Kỹ năng cần có: [5]**

- **Kiến thức chuyên môn:**

- + Thành thạo ngôn ngữ lập trình Java.
- + Hiểu biết về các framework Java phổ biến như Spring Boot, Hibernate.
- + Kiến thức về HTML, CSS và JavaScript.
- + Kiến thức về cơ sở dữ liệu SQL.
- + Hiểu biết về các nguyên tắc thiết kế phần mềm hướng đối tượng.

- **Kỹ năng mềm:**

- + Khả năng giao tiếp và làm việc nhóm tốt.
- + Khả năng học hỏi và thích nghi nhanh chóng.
- + Khả năng giải quyết vấn đề và tư duy logic tốt.
- + Chú ý đến chi tiết và có tinh thần trách nhiệm cao.



## CHƯƠNG 2. TỔNG QUAN VỀ LÝ THUYẾT

### 2.1. Giới thiệu về private Framework HNV-NewLine và SpringBoot.

#### 2.1.1 Private Framework HNV-NewLine

HNV-NewLine là một framework private được phát triển bởi HNV Corporation nhằm hỗ trợ các dự án Java Enterprise. Framework này cung cấp một bộ thư viện và công cụ giúp các nhà phát triển xây dựng ứng dụng Java Enterprise một cách nhanh chóng, hiệu quả và dễ dàng.

HNV-NewLine được xây dựng dựa trên nền tảng Spring Boot, một framework Java phổ biến và mạnh mẽ. Spring Boot cung cấp một loạt các tính năng giúp đơn giản hóa việc phát triển ứng dụng Java Enterprise, bao gồm:

- + Khởi động ứng dụng dễ dàng mà không cần cấu hình XML.
- + Tự động cấu hình các bean Spring.
- + Hỗ trợ Dependency Injection.
- + Hỗ trợ Spring MVC để xây dựng RESTful API.
- + Hỗ trợ Spring Data JPA để truy cập cơ sở dữ liệu.

HNV-NewLine bổ sung thêm các tính năng và công cụ sau đây cho Spring Boot:

- + Cấu trúc mã chuẩn hóa: HNV-NewLine cung cấp một cấu trúc mã chuẩn hóa cho các ứng dụng Java Enterprise. Cấu trúc này giúp các nhà phát triển dễ dàng hiểu và duy trì codebase.

- + Bộ thư viện tiện ích: HNV-NewLine cung cấp một bộ thư viện tiện ích giúp các nhà phát triển thực hiện các tác vụ chung trong các ứng dụng Java Enterprise, chẳng hạn như:

- Xử lý lỗi
- Xác thực và ủy quyền
- Quản lý tài nguyên
- Giao tiếp với các dịch vụ bên ngoài

Công cụ phát triển: HNV-NewLine cung cấp một số công cụ phát triển giúp các nhà phát triển làm việc hiệu quả hơn, chẳng hạn như:

- + Công cụ tạo mã
- + Công cụ gỡ lỗi
- + Công cụ kiểm tra

### 2.1.2 SpringBoot. [6]

SpringBoot là một framework Java phổ biến và mạnh mẽ được sử dụng để phát triển các ứng dụng web. SpringBoot cung cấp một loạt các tính năng giúp đơn giản hóa việc phát triển ứng dụng Java Enterprise, bao gồm:

- + Khởi động ứng dụng dễ dàng mà không cần cấu hình XML
- + Tự động cấu hình các bean Spring
- + Hỗ trợ Dependency Injection
- + Hỗ trợ Spring MVC để xây dựng RESTful API
- + Hỗ trợ Spring Data JPA để truy cập cơ sở dữ liệu

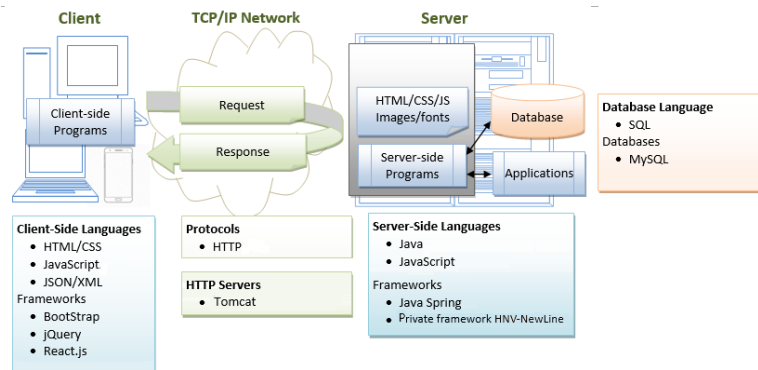
SpringBoot có một số ưu điểm sau:

- + Dễ sử dụng: SpringBoot rất dễ sử dụng và học hỏi. Các nhà phát triển có thể bắt đầu xây dựng các ứng dụng SpringBoot trong vài phút.
- + Nhanh chóng: SpringBoot giúp các nhà phát triển xây dựng ứng dụng nhanh chóng và hiệu quả. SpringBoot cung cấp một loạt các tính năng giúp tự động hóa nhiều tác vụ chung trong phát triển ứng dụng Java Enterprise.
- + Mạnh mẽ: SpringBoot là một framework mạnh mẽ và có thể mở rộng. SpringBoot có thể được sử dụng để xây dựng các ứng dụng Java Enterprise lớn và phức tạp.



Hình 2.1. Đặc điểm của Sprint Boot

## 2.2. Tìm hiểu về các kiến trúc phần mềm phù hợp cho ứng dụng quản lý dự án.



Hình 2.2 Kiến trúc ứng dụng web zino sử dụng.

### 2.2.1 Kiến trúc nhiều lớp (Multi-tier architecture). [7]

Kiến trúc nhiều lớp là một kiến trúc phần mềm phổ biến được sử dụng cho các ứng dụng web phức tạp. Kiến trúc này chia ứng dụng thành ba lớp:

- + Lớp trình bày (Presentation layer): Lớp này chịu trách nhiệm tương tác với người dùng. Nó bao gồm các thành phần như giao diện người dùng web, giao diện người dùng di động và API.
- + Lớp dịch vụ (Business logic layer): Lớp này chịu trách nhiệm thực hiện logic kinh doanh của ứng dụng. Nó bao gồm các thành phần như dịch vụ web, dịch vụ RESTful và các quy trình làm việc.
- + Lớp truy cập dữ liệu (Data access layer): Lớp này chịu trách nhiệm truy cập và quản lý dữ liệu của ứng dụng. Nó bao gồm các thành phần như cơ sở dữ liệu, hệ thống quản lý cơ sở dữ liệu và các truy vấn SQL.

Kiến trúc nhiều lớp có một số ưu điểm cho các ứng dụng quản lý dự án, bao gồm:

- + Khả năng mở rộng: Kiến trúc nhiều lớp có thể dễ dàng mở rộng để đáp ứng nhu cầu của các nhóm và tổ chức quy mô lớn.
- + Bảo trì: Kiến trúc nhiều lớp dễ bảo trì hơn vì các lớp được tách biệt khỏi nhau.
- + Khả năng tái sử dụng: Các thành phần trong kiến trúc nhiều lớp có thể được tái sử dụng trong các ứng dụng khác.

### **2.2.2 Kiến trúc vi mô dịch vụ (Microservices architecture) [8]**

Kiến trúc vi mô dịch vụ là một kiến trúc phần mềm mới nổi được sử dụng cho các ứng dụng phức tạp. Kiến trúc này chia ứng dụng thành các dịch vụ nhỏ, độc lập. Mỗi dịch vụ có trách nhiệm thực hiện một chức năng cụ thể và có thể được triển khai và quản lý riêng biệt.

Kiến trúc vi mô dịch vụ có một số ưu điểm cho các ứng dụng quản lý dự án, bao gồm:

- + Khả năng mở rộng: Kiến trúc vi mô dịch vụ có thể dễ dàng mở rộng bằng cách thêm các dịch vụ mới.
- + Linh hoạt: Kiến trúc vi mô dịch vụ linh hoạt hơn kiến trúc nhiều lớp vì các dịch vụ có thể được triển khai và quản lý riêng biệt.
- + Khả năng phục hồi: Kiến trúc vi mô dịch vụ có khả năng phục hồi cao hơn vì nếu một dịch vụ bị lỗi, các dịch vụ khác vẫn có thể hoạt động.

### **2.2.3 Kiến trúc Hexagonal (Hexagonal architecture). [9]**

Kiến trúc Hexagonal là một kiến trúc phần mềm tập trung vào việc tách biệt logic kinh doanh khỏi các chi tiết phụ thuộc bên ngoài. Kiến trúc này sử dụng các cổng và adapter để giao tiếp với các hệ thống bên ngoài.

Kiến trúc Hexagonal có một số ưu điểm cho các ứng dụng quản lý dự án, bao gồm:

- + Khả năng kiểm thử: Kiến trúc Hexagonal dễ kiểm thử hơn vì logic kinh doanh có thể được kiểm thử mà không cần phụ thuộc vào các hệ thống bên ngoài.

- + Bảo trì: Kiến trúc Hexagonal dễ bảo trì hơn vì logic kinh doanh được tách biệt khỏi các chi tiết phụ thuộc bên ngoài.
- + Khả năng tái sử dụng: Logic kinh doanh trong kiến trúc Hexagonal có thể được tái sử dụng trong các ứng dụng khác.

## **2.3. Các mô hình dữ liệu và quản lý dữ liệu trong ứng dụng quản lý dự án.**

### **2.3.1 Các mô hình dữ liệu trong Zino.**

Zino sử dụng nhiều mô hình dữ liệu khác nhau để lưu trữ thông tin về các dự án, nhiệm vụ, người dùng và các thành phần khác của ứng dụng. Một số mô hình dữ liệu phổ biến bao gồm:

- + Mô hình dữ liệu dự án: Lưu trữ thông tin về dự án, chẳng hạn như tên dự án, mô tả, ngày bắt đầu, ngày kết thúc, trạng thái dự án, ...
- + Mô hình dữ liệu nhiệm vụ: Lưu trữ thông tin về các nhiệm vụ trong dự án, chẳng hạn như tên nhiệm vụ, mô tả, người được giao, ngày đến hạn, trạng thái nhiệm vụ, ...
- + Mô hình dữ liệu người dùng: Lưu trữ thông tin về người dùng của ứng dụng, chẳng hạn như tên người dùng, mật khẩu, email, vai trò người dùng, ...
- + Mô hình dữ liệu tệp đính kèm: Lưu trữ thông tin về các tệp được đính kèm vào dự án hoặc nhiệm vụ, chẳng hạn như tên tệp, kích thước tệp, loại tệp, ...

### **2.3.2 Quản lý dữ liệu trong Zino.**

Zino sử dụng nhiều kỹ thuật quản lý dữ liệu khác nhau để đảm bảo tính toàn vẹn, nhất quán và khả năng truy cập của dữ liệu. Một số kỹ thuật phổ biến bao gồm:

- + Kiểm soát truy cập: Zino sử dụng kiểm soát truy cập để hạn chế quyền truy cập vào dữ liệu. Điều này có nghĩa là chỉ những người dùng có quyền thích hợp mới có thể xem, sửa đổi hoặc xóa dữ liệu.
- + Sao lưu dữ liệu: Zino sao lưu dữ liệu thường xuyên để đảm bảo rằng dữ liệu có thể được khôi phục trong trường hợp xảy ra sự cố.

- + Phục hồi dữ liệu: Zino có các quy trình phục hồi dữ liệu để khôi phục dữ liệu trong trường hợp bị xóa hoặc bị hỏng.
- + Bảo mật dữ liệu: Zino sử dụng các biện pháp bảo mật khác nhau để bảo vệ dữ liệu khỏi truy cập trái phép, sử dụng và tiết lộ.

## **2.4. Các phương pháp bảo mật cho ứng dụng web.**

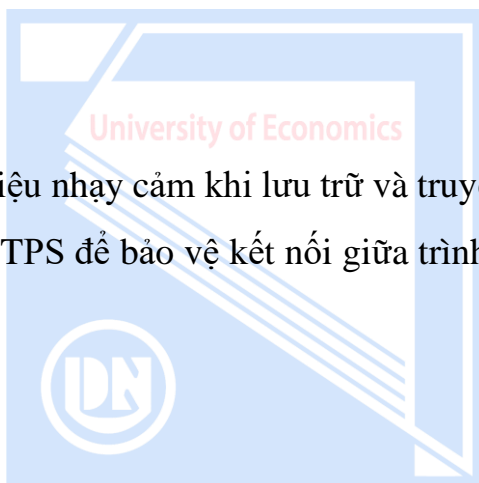
### **2.4.1 Xác thực và ủy quyền.**

**Xác thực:** Xác minh danh tính của người dùng khi họ truy cập ứng dụng web. Các phương pháp xác thực phổ biến bao gồm sử dụng tên người dùng và mật khẩu.

**Ủy quyền:** Xác định quyền truy cập của người dùng đối với các tài nguyên và chức năng của ứng dụng web. Việc ủy quyền dựa trên vai trò của người dùng hoặc các thuộc tính khác.

### **2.4.2 Mã hóa dữ liệu.**

- + Mã hóa dữ liệu nhạy cảm khi lưu trữ và truyền tải.
- + Sử dụng HTTPS để bảo vệ kết nối giữa trình duyệt của người dùng và máy chủ web.



## CHƯƠNG 3. GIỚI THIỆU DỰ ÁN VÀ PHẦN MỀM.

### 3.1. Giới thiệu về dự án quản lý dự án Zino.

#### 3.1.1 Tổng quan về dự án Zino.

Hiện nay, quản lý dự án được coi là công việc rất quan trọng trong việc đảm bảo cho các chiến lược và kế hoạch của doanh nghiệp được hoàn thành đúng tiêu chuẩn và trong thời gian quy định. Đây là một công việc không thể thiếu đối với bất kỳ doanh nghiệp nào. Vậy nên phần mềm quản lý dự án đã trở thành công cụ không thể thiếu cho các nhà quản trị. Trong quá trình phát triển sản phẩm, hệ thống đảm bảo dự án hoàn thành đúng tiến độ và ngân sách, đồng thời giảm thiểu rủi ro tối đa.

Zino là một phần mềm để quản trị nhiều dự án trong doanh nghiệp, giúp quản lý công việc hiệu quả hơn, từ việc lập kế hoạch, phân công nhiệm vụ đến theo dõi tiến độ và đánh giá hiệu suất của từng thành viên/ nhân viên trong tổ chức. Kết nối công việc và sự hợp tác của các thành viên trong team phát triển và các team khác nhiều hơn trong tổ chức, thông qua các tính năng phòng trò chuyện hay bình luận và chia sẻ tài liệu. Ngoài ra, hệ thống giúp các nhà quản lý dự án báo cáo, phân tích, đánh giá hiệu quả làm việc trong từng dự án thông qua số liệu mà hệ thống tổng hợp lại

#### 3.1.2 Lợi ích của dự án Zino.

**Quản lý dự án hiệu quả:** Zino giúp bạn lập kế hoạch chi tiết cho dự án, chia nhỏ thành các nhiệm vụ cụ thể, phân công nhiệm vụ cho các thành viên trong nhóm, theo dõi tiến độ thực hiện và đánh giá hiệu quả công việc. Nhờ vậy, có thể đảm bảo dự án được hoàn thành đúng tiến độ, ngân sách và chất lượng đề ra.

**Tăng cường sự hợp tác:** Zino cung cấp các tính năng cộng tác mạnh mẽ giúp các thành viên trong nhóm dễ dàng giao tiếp, chia sẻ thông tin và cập nhật tiến độ công việc. Nhờ vậy, mọi người luôn được cập nhật thông tin mới nhất và có thể phối hợp hiệu quả để hoàn thành dự án.

**Cải thiện hiệu suất:** Zino giúp bạn theo dõi hiệu suất làm việc của từng thành viên trong nhóm và xác định những điểm khó khăn có thể ảnh hưởng đến tiến độ dự án. Nhờ vậy, bạn có thể đưa ra các điều chỉnh kịp thời để cải thiện hiệu quả công việc và đảm bảo dự án được hoàn thành đúng hạn.

**Giảm thiểu rủi ro:** Zino giúp bạn xác định và đánh giá các rủi ro tiềm ẩn có thể ảnh hưởng đến dự án. Nhờ vậy, bạn có thể xây dựng các biện pháp phòng ngừa và giảm thiểu rủi ro, giúp dự án diễn ra suôn sẻ hơn.

**Tiết kiệm thời gian và chi phí:** Zino tự động hóa nhiều quy trình thủ công, giúp bạn tiết kiệm thời gian và chi phí. Nhờ vậy, bạn có thể tập trung vào những công việc quan trọng hơn và nâng cao hiệu quả tổng thể của dự án..

### 3.1.3 Chức năng chính của dự án.

Quản lý dự án:

- + Tạo và quản lý dự án.
- + Chia nhỏ dự án thành các nhiệm vụ cụ thể.
- + Phân công nhiệm vụ cho các thành viên trong nhóm.
- + Theo dõi tiến độ thực hiện nhiệm vụ.

Đánh giá hiệu quả công việc:

- + Quản lý rủi ro.
- + Lập báo cáo và phân tích dữ liệu dự án.

Hợp tác:

- + Trò chuyện nhóm và riêng tư.
- + Gửi bình luận và ghi chú cho các nhiệm vụ.
- + Chia sẻ tệp tin và tài liệu.
- + @ đề cập đến các thành viên trong nhóm.
- + Theo dõi hoạt động của thành viên trong nhóm.

Lịch trình:



- + Lập lịch trình thực hiện dự án theo từng giai đoạn.
- + Tạo các lịch hẹn cho các cuộc họp, báo cáo,....
- + Theo dõi tiến độ thực hiện lịch trình.
- + Điều chỉnh lịch trình khi cần thiết.
- + Gửi thông báo về các thay đổi lịch trình cho các thành viên trong nhóm.

#### Lưu trữ:

- + Lưu trữ tất cả các tài liệu liên quan đến dự án ở một nơi tập trung.
- + Tìm kiếm tài liệu dễ dàng.
- + Chia sẻ tài liệu với các thành viên trong nhóm.
- + Kiểm soát quyền truy cập vào tài liệu.

#### Báo cáo:

- + Tạo các báo cáo về tiến độ dự án, hiệu suất công việc, rủi ro và các thông tin khác.
- + Lọc và tùy chỉnh báo cáo theo nhu cầu.
- + Xuất báo cáo sang các định dạng khác nhau như PDF, Excel, Word, ....

#### Bảo mật:

- + Bảo vệ dữ liệu dự án bằng các biện pháp bảo mật tiên tiến.
- + Quyền truy cập vào dự án được kiểm soát chặt chẽ.
- + Theo dõi hoạt động của người dùng.

#### Tích hợp:

- + Tích hợp với các công cụ khác như chat, email,...
- + Tự động hóa các quy trình làm việc.
- + Nâng cao hiệu quả hoạt động của doanh nghiệp.

### **3.2. Giới thiệu Phần mềm Eclipse và ngôn ngữ lập trình Java, cùng các công cụ khác.**

#### **3.2.1 Giới thiệu Phần mềm Eclipse. [10]**



Hình 3.1. Eclipse.

Eclipse là một môi trường phát triển tích hợp (IDE) mã nguồn mở phổ biến được sử dụng cho lập trình máy tính. Nó được viết chủ yếu bằng Java và hỗ trợ nhiều ngôn ngữ lập trình khác nhau, bao gồm Java, C/C++, PHP, Ruby, và nhiều ngôn ngữ khác. Eclipse cung cấp một bộ công cụ toàn diện cho các nhà phát triển, bao gồm:

- + Biên tập mã: Eclipse cung cấp một trình soạn thảo mã mạnh mẽ với nhiều tính năng như tô sáng cú pháp, tự động hoàn thành mã, và kiểm tra lỗi cú pháp.
- + Công cụ gỡ lỗi: Eclipse bao gồm một trình gỡ lỗi tích hợp giúp các nhà phát triển theo dõi và sửa lỗi trong mã của họ.
- + Công cụ quản lý dự án: Eclipse cung cấp các công cụ để quản lý dự án phát triển phần mềm, bao gồm theo dõi lỗi, quản lý tác vụ và kiểm soát phiên bản.
- + Khả năng mở rộng: Eclipse có thể được mở rộng bằng các plugin, cung cấp thêm chức năng cho IDE. Có hàng ngàn plugin có sẵn cho Eclipse, bao gồm plugin cho các ngôn ngữ lập trình khác nhau, khung phần mềm và công cụ phát triển.

Eclipse là một lựa chọn phổ biến cho các nhà phát triển Java vì nó cung cấp một bộ công cụ toàn diện và dễ sử dụng để phát triển ứng dụng Java. Eclipse cũng là một nền tảng phổ biến cho các dự án Java mã nguồn mở.

### 3.2.2 Ngôn ngữ lập trình Java. [11]



Hình 3.2. Java

Java là ngôn ngữ lập trình hướng đối tượng phổ biến được sử dụng để phát triển nhiều loại ứng dụng, bao gồm ứng dụng web, ứng dụng di động, ứng dụng doanh nghiệp và ứng dụng khoa học. Java được biết đến với tính bảo mật, độ tin cậy và khả năng di động cao. Một số tính năng chính của Java bao gồm:

- + Hướng đối tượng: Java là ngôn ngữ hướng đối tượng, có nghĩa là nó cho phép lập trình viên tạo ra các đối tượng, là các khối xây dựng cơ bản của phần mềm hướng đối tượng.
- + Quản lý bộ nhớ tự động: Java có tính năng quản lý bộ nhớ tự động, có nghĩa là lập trình viên không cần phải lo lắng về việc phân bổ và giải phóng bộ nhớ thủ công.
- + Kiểm tra loại tĩnh: Java có tính năng kiểm tra loại tĩnh, giúp xác định lỗi cú pháp và logic trong mã trước khi chương trình được thực thi.
- + Khả năng di động: Java là ngôn ngữ lập trình di động, có nghĩa là các chương trình Java có thể được thực thi trên bất kỳ nền tảng nào có Máy ảo Java (JVM).

Java là một lựa chọn phổ biến cho các nhà phát triển phần mềm vì nó là ngôn ngữ mạnh mẽ, linh hoạt và dễ sử dụng. Java cũng là một ngôn ngữ phổ biến cho các doanh nghiệp vì nó là ngôn ngữ bảo mật và đáng tin cậy.

### 3.2.3 MySQL. [12]

MySQL được phát triển vào năm 1994, MySQL là một hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở được hỗ trợ bởi Oracle dựa trên Ngôn ngữ truy vấn có cấu trúc (SQL), theo máy khách – máy chủ (mô hình client-server).

Cách thức hoạt động:

- + MySQL hoạt động dựa trên mô hình client – server. Cốt lõi của MySQL là máy chủ MySQL, xử lý tất cả các hướng dẫn CSDL hoặc các lệnh. Máy chủ SQL có sẵn như một chương trình riêng biệt để sử dụng trong môi trường mạng client –

server. Nó như một thư viện có thể được nhúng hoặc liên kết vào các ứng dụng riêng biệt.

+ MySQL hoạt động cùng một số chương trình tiện ích hỗ trợ quản trị CSDL MySQL. Các lệnh được gửi đến MySQL Server thông qua máy khách được cài đặt trên máy tính.



## CHƯƠNG 4. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

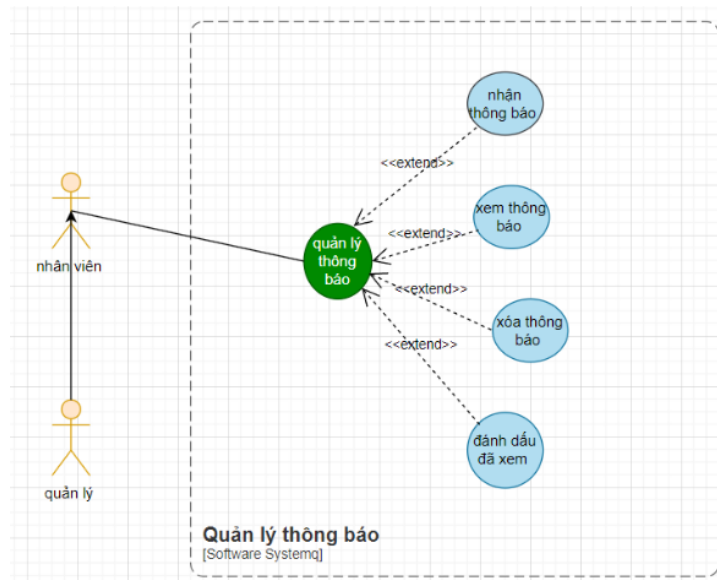
### 4.1. Phân tích yêu cầu hệ thống

#### 4.1.1 Đặc tả các trường hợp yêu cầu thông báo.

- + Thông báo tạo mới thành công một dự án/ nhiệm vụ mới.
- + Thông báo được thêm vào làm thành viên của dự án, nhiệm vụ, lịch hẹn.
- + Sự kiện quan trọng sắp tới: Hệ thống có thể gửi thông báo để nhắc nhở các thành viên về các sự kiện quan trọng sắp tới như:
  - Lịch hẹn sắp tới: Nhắc nhở các thành viên có trong lịch hẹn.
  - Nhắc nhở sắp hết hạn phân việc: Nếu một nhiệm vụ chưa hoàn thành mà thời hạn sắp tới, hệ thống cần gửi thông báo nhắc nhở cho thành viên phụ trách và quản lý dự án.
  - Nhiệm vụ quá hạn: Nếu một nhiệm vụ không được hoàn thành trước thời hạn, hệ thống cần gửi thông báo nhắc nhở cho thành viên phụ trách và quản lý dự án.
- + Thay đổi trạng thái nhiệm vụ: Khi trạng thái của một nhiệm vụ thay đổi (ví dụ: từ "Đang thực hiện" sang "Hoàn thành"), hệ thống có thể gửi thông báo cho các bên liên quan.
- + Phản hồi hoặc nhận xét mới: Nếu có phản hồi hoặc nhận xét mới trên một nhiệm vụ hoặc dự án, hệ thống có thể gửi thông báo để những người liên quan có thể theo dõi và phản hồi kịp thời.
- + Cập nhật dự án: Khi có cập nhật quan trọng về dự án, như thay đổi ngân sách, thời hạn, hoặc phạm vi công việc, hệ thống có thể gửi thông báo cho tất cả các thành viên dự án.
- + Nhắc nhở định kỳ: Hệ thống có thể gửi thông báo nhắc nhở định kỳ về các nhiệm vụ đang thực hiện, thời hạn sắp tới, hoặc các hoạt động khác cần theo dõi.
- + Thêm mới lịch hẹn: Khi tạo một lịch hẹn thì hệ thống sẽ gửi thông báo cho các thành viên có trong lịch hẹn.

- + Thêm mới bài viết: Hệ thống sẽ tạo thông báo và gửi thông báo tới cho các admin có quyền duyệt bài viết.
- + Duyệt bài viết: Sau khi admin duyệt bài viết thì hệ thống sẽ gửi thông báo tới thành viên tạo bài viết.

#### 4.1.2 Use case



Hình 4.1. Use case quản lý thông báo

#### 4.1.3 Đặc tả use case

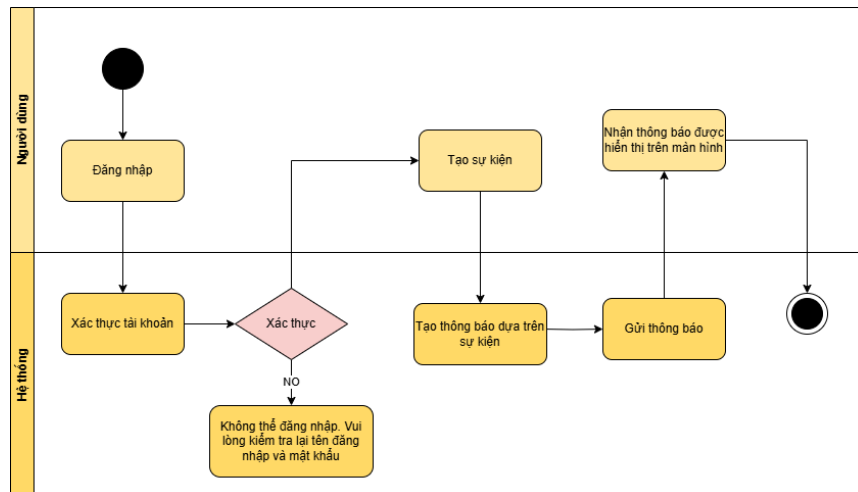
##### a. *Đặc tả chi tiết use case nhận thông báo.*

Use case ID	1
Use case name	Nhận thông báo
Description	Là người dùng, tôi muốn nhận được thông báo khi có các sự kiện quan trọng xảy ra trong hệ thống quản lý dự án như tạo dự án mới, thêm thành viên vào dự án, giao công việc mới, thay đổi trạng thái công việc, nhận phản hồi hoặc cập nhật quan trọng về dự án.
Actors	Quản lý, nhân viên
Priority	High

Triggers	Khi có một sự kiện quan trọng xảy ra trong dự án (tạo dự án mới, thêm thành viên vào dự án, gán công việc mới, thay đổi trạng thái công việc, nhận phản hồi hoặc cập nhật quan trọng về dự án) nhận được thông báo
Pre-conditions	<ul style="list-style-type: none"> <li>• Người dùng đã đăng nhập vào hệ thống.</li> <li>• Người dùng là thành viên của dự án hoặc có liên quan đến công việc trong dự án.</li> </ul>
Post-conditions	<ul style="list-style-type: none"> <li>• Thông báo được hiển thị cho người dùng trên giao diện hệ thống.</li> <li>• Người dùng có thể xem chi tiết thông báo và thực hiện các hành động liên quan.</li> </ul>
Main flow	<ol style="list-style-type: none"> <li>1. Sự kiện quan trọng xảy ra trong dự án.</li> <li>2. Hệ thống tạo một thông báo mới dựa trên sự kiện này.</li> <li>3. Hệ thống gửi thông báo đến người dùng có liên quan.</li> <li>4. Người dùng nhận được thông báo trên giao diện hệ thống.</li> </ol>
Alternative flows	4a. Nếu người dùng không đăng nhập, thông báo sẽ được lưu lại và hiển thị khi người dùng đăng nhập lần sau.
Exception flows	3a. Nếu có lỗi trong quá trình gửi thông báo, hệ thống ghi lại lỗi và thử gửi lại sau một khoảng thời gian định trước.
Business rules	<ul style="list-style-type: none"> <li>• Thông báo chỉ được gửi đến những người dùng có liên quan đến sự kiện.</li> <li>• Thông báo phải được gửi trong thời gian thực hoặc gần thời gian thực.</li> <li>• Thông báo phải rõ ràng, chính xác và có liên kết đến chi tiết sự kiện nếu cần.</li> </ul>

Non-functional requirements	<ul style="list-style-type: none"> <li>Thông báo phải được gửi trong vòng 5 giây kể từ khi sự kiện xảy ra.</li> <li>Hệ thống phải có khả năng xử lý ít nhất 1000 thông báo mỗi giây.</li> <li>Giao diện thông báo phải thân thiện và dễ sử dụng.</li> </ul>
-----------------------------	---

Bảng 4.1. Đặc tả chi tiết use case nhận thông báo.



Hình 4.2. Luồng nhận thông báo.

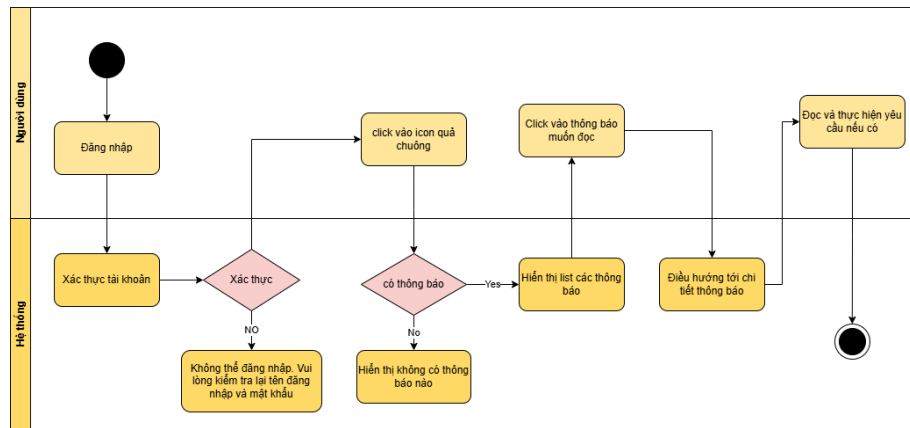
**b. Đặc tả chi tiết use case xem thông báo**

Use case ID	2
Use case name	Xem thông báo.
Description	Là người dùng, tôi muốn xem các thông báo đã nhận được về các sự kiện quan trọng xảy ra trong hệ thống quản lý dự án.
Actors	Quản lý, nhân viên
Priority	High
Triggers	<ul style="list-style-type: none"> <li>Người dùng chọn xem danh sách thông báo trên giao diện hệ thống.</li> <li>Người dùng nhận được thông báo mới và chọn xem chi tiết.</li> </ul>



Pre-conditions	<ul style="list-style-type: none"> <li>• Người dùng đã đăng nhập vào hệ thống.</li> <li>• Người dùng đã nhận được ít nhất một thông báo.</li> </ul>
Post-conditions	<ul style="list-style-type: none"> <li>• Thông báo được đánh dấu là đã xem.</li> <li>• Người dùng có thể thực hiện các hành động liên quan từ thông báo (nếu có).</li> </ul>
Main flow	<ol style="list-style-type: none"> <li>1. Chọn biểu tượng thông báo trên giao diện hệ thống.</li> <li>2. Hệ thống hiển thị danh sách các thông báo.</li> <li>3. Chọn một thông báo từ danh sách.</li> <li>4. Hệ thống hiển thị chi tiết thông báo.</li> <li>5. Đọc và thực hiện các hành động liên quan từ thông báo (nếu có).</li> </ol>
Alternative flows	<b>2a.</b> Nếu không có thông báo nào, hệ thống hiển thị thông báo "Không có thông báo mới".
Exception flows	<p><b>3a.</b> Nếu có lỗi trong việc tải danh sách thông báo, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng thử lại.</p> <p><b>4a.</b> Nếu có lỗi trong việc tải chi tiết thông báo, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng thử lại.</p>
Business rules	<ul style="list-style-type: none"> <li>• Thông báo mới phải được hiển thị trên cùng của danh sách.</li> <li>• Thông báo chưa xem phải được đánh dấu rõ ràng để người dùng dễ nhận biết.</li> </ul>
Non-functional requirements	<ul style="list-style-type: none"> <li>• Thời gian tải danh sách thông báo không quá 3 giây.</li> <li>• Thời gian tải chi tiết thông báo không quá 2 giây.</li> <li>• Giao diện hiển thị thông báo phải rõ ràng, dễ hiểu</li> </ul>

Bảng 4.2. Đặc tả chi tiết use case xem thông báo.



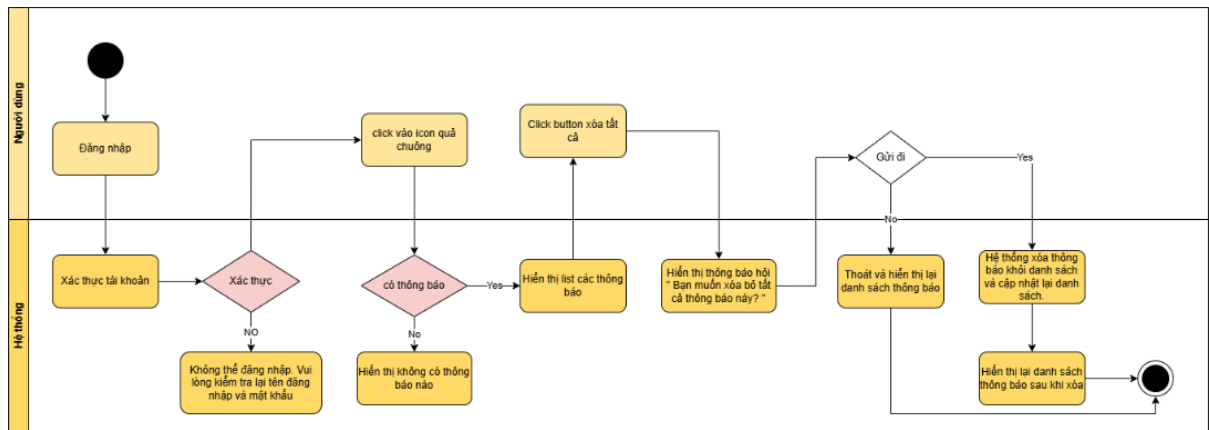
Hình 4.3. Luồng xem thông báo

**c. Đặc tả chi tiết use case xóa thông báo**

Use case ID	3
Use case name	Xóa thông báo.
Description	Là người dùng, tôi muốn xóa một hoặc nhiều thông báo mà tôi đã nhận được từ hệ thống quản lý dự án.
Actors	Quản lý, nhân viên
Priority	Medium
Triggers	<ul style="list-style-type: none"> <li>Người dùng chọn xóa một thông báo từ danh sách thông báo.</li> <li>Người dùng chọn xóa tất cả thông báo.</li> </ul>
Pre-conditions	<ul style="list-style-type: none"> <li>Người dùng đã đăng nhập vào hệ thống.</li> <li>Người dùng đã nhận được ít nhất một thông báo.</li> </ul>
Post-conditions	<ul style="list-style-type: none"> <li>Thông báo được xóa khỏi danh sách thông báo của người dùng.</li> <li>Danh sách thông báo được cập nhật và hiển thị lại cho người dùng.</li> </ul>
Main flow	<ol style="list-style-type: none"> <li>Chọn biểu tượng thông báo trên giao diện hệ thống.</li> <li>Hệ thống hiển thị danh sách các thông báo.</li> <li>Chọn thông báo cần xóa.</li> <li>Xác nhận xóa thông báo.</li> </ol>

	<p>5. Hệ thống xóa thông báo khỏi danh sách và cập nhật lại danh sách.</p> <p>6. Danh sách thông báo đã được cập nhật.</p>
Alternative flows	<p>3a. Nếu người dùng chọn "Xóa tất cả thông báo":</p> <ul style="list-style-type: none"> <li>• Hệ thống yêu cầu người dùng xác nhận hành động này.</li> <li>• Người dùng xác nhận xóa tất cả thông báo.</li> <li>• Hệ thống xóa tất cả thông báo khỏi danh sách và cập nhật lại danh sách.</li> <li>• Người dùng thấy danh sách thông báo trống.</li> </ul>
Exception flows	<p>4a. Nếu có lỗi trong việc xóa thông báo, hệ thống hiển thị thông báo lỗi và yêu cầu người dùng thử lại.</p> <p>4b. Nếu người dùng hủy bỏ việc xóa thông báo, hệ thống quay lại danh sách thông báo mà không có thay đổi.</p>
Business rules	<ul style="list-style-type: none"> <li>• Thông báo xóa phải được loại bỏ hoàn toàn khỏi hệ thống và không thể khôi phục.</li> <li>• Người dùng có thể chọn xóa một hoặc nhiều thông báo cùng lúc.</li> <li>• Hệ thống phải xác nhận lại trước khi xóa tất cả thông báo để tránh nhầm lẫn.</li> </ul>
Non-functional requirements	<ul style="list-style-type: none"> <li>• Thời gian xử lý việc xóa thông báo không quá 2 giây.</li> <li>• Giao diện xóa thông báo phải rõ ràng, dễ sử dụng và yêu cầu xác nhận từ người dùng khi xóa tất cả thông báo</li> </ul>

Bảng 4.3. Đặc tả chi tiết use case xóa thông báo.



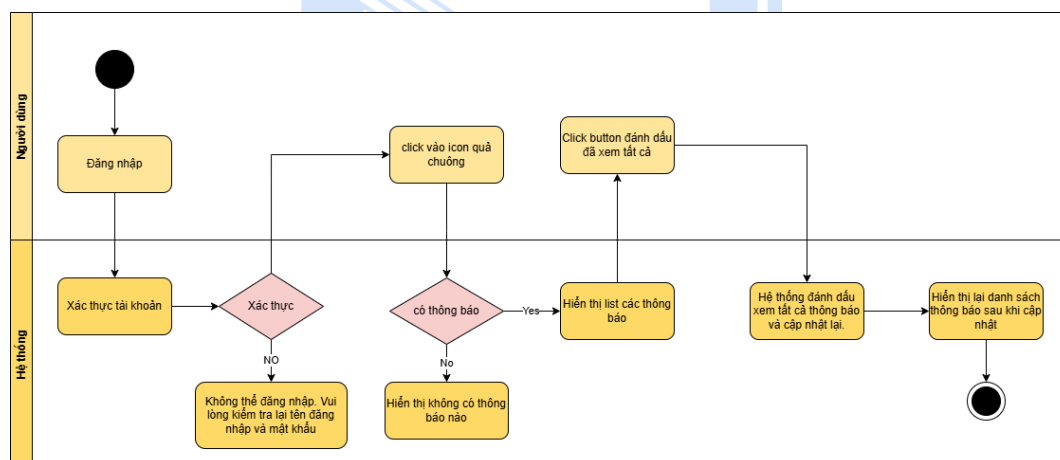
Hình 4.4. Luồng xóa thông báo.

d. **Đặc tả chi tiết use case đánh dấu đã xem**

Use case ID	2
Use case name	Xem thông báo.
Description	Là người dùng, tôi muốn đánh dấu tất cả thông báo đã xem để quản lý và tổ chức thông báo một cách hiệu quả.
Actors	Quản lý, nhân viên
Priority	High
Triggers	<ul style="list-style-type: none"> <li>Chọn đánh dấu tất cả thông báo là đã xem từ danh sách thông báo.</li> </ul>
Pre-conditions	<ul style="list-style-type: none"> <li>Người dùng đã đăng nhập vào hệ thống.</li> <li>Người dùng đã nhận được ít nhất một thông báo.</li> </ul>
Post-conditions	<ul style="list-style-type: none"> <li>Thông báo được đánh dấu là đã xem.</li> <li>Danh sách thông báo được cập nhật và hiển thị lại cho người dùng với các thông báo đã xem được đánh dấu tương ứng.</li> </ul>
Main flow	<ol style="list-style-type: none"> <li>Chọn biểu tượng thông báo trên giao diện hệ thống.</li> <li>Hệ thống hiển thị danh sách các thông báo.</li> <li>Chọn nút đánh dấu là đã xem.</li> <li>Hệ thống đánh dấu thông báo là đã xem và cập nhật lại danh sách thông báo.</li> </ol>

	5. Người dùng thấy danh sách thông báo đã được cập nhật, thông báo đã xem được đánh dấu tương ứng.
Alternative flows	N/A
Exception flows	N/A
Business rules	<ul style="list-style-type: none"> <li>Thông báo đã xem phải được hiển thị khác biệt so với thông báo chưa xem (ví dụ: màu sắc, biểu tượng).</li> </ul>
Non-functional requirements	<ul style="list-style-type: none"> <li>Thời gian xử lý việc đánh dấu thông báo là đã xem không quá 2 giây.</li> <li>Giao diện đánh dấu thông báo đã xem phải rõ ràng, dễ sử dụng.</li> </ul>

Bảng 4.4. Đặc tả chi tiết use case đánh dấu đã xem thông báo.



Hình 4.5. Luồng đánh dấu đã xem

## 4.2. Thiết kế cơ sở dữ liệu

### 4.2.1 Mức khái niệm

STT	Viết tắt	Mô tả dữ liệu
1	I_ID	ID của thông báo/ message
2	I_Status	ID của các trạng thái được mặc định trong chương trình code

3	I_Type_01	Loại loại thông điệp được gửi type Msg: 1:email, 2:chat
4	I_Type_02	Loại thông báo được gửi đến người dùng. Các loại thông báo này có thể bao gồm: <ul style="list-style-type: none"> <li>• <b>SMS (envoi sms)</b>: Gửi tin nhắn văn bản qua dịch vụ SMS.</li> <li>• <b>Email (envoi email)</b>: Gửi thông báo qua email.</li> <li>• <b>In-app</b>: Gửi thông báo trong ứng dụng, tức là người dùng nhận thông báo trực tiếp trong giao diện của ứng dụng mà họ đang sử dụng.</li> </ul>
5	T_Info_01	Thông tin về người gửi tin nhắn, email, lưu trữ nội dung của các thông báo
6	T_Info_02	Thông tin về người nhận tin nhắn. email, thông báo.
7	T_Info_03	Lưu trữ nội dung của tin nhắn/ email
8	T_Info_04	Lưu trữ nội dung của tin nhắn/ email
9	T_Info_05	Lưu trữ các thông tin bổ sung
10	I_Aut_User	ID của người dùng tạo sự kiện thông báo hoặc gửi tin nhắn/ email
11	D_Date_01	Ngày khởi tạo
12	D_Date_02	Ngày sửa đổi . Nếu nội dung được chỉnh sửa, cột này sẽ ghi lại thời điểm sửa đổi, giúp theo dõi lịch sử chỉnh sửa.
13	I_Entity_Type	Loại thực thể mà thông điệp liên quan đến. Ví dụ: 1 có thể là dự án, 2 là nhiệm vụ. Điều này giúp phân loại và quản lý các thông điệp theo các thực thể khác nhau trong hệ thống.
14	I_Entity_ID	Chứa id của dự án hoặc nhiệm vụ

Bảng 4.5. CSDL mức khái niệm

#### 4.2.2 Mức logic

STT	Thuộc tính	Ràng buộc	Kiểu dữ liệu
1	I_ID	Primary Key	int(11)
2	I_Status		int(11)

3	I_Type_01		int(11)
4	I_Type_02		int(11)
5	T_Info_01		varchar(2000)
6	T_Info_02		varchar(2000)
7	T_Info_03		varchar(2000)
8	T_Info_04		longtext
9	T_Info_05		longtext
10	I_Aut_User		int(11)
11	D_Date_01		datetime
12	D_Date_02		datetime
13	I_Entity_Type		int(11)
14	I_Entity_ID		int(11)

Bảng 4.6. CSDL mức logic

### 4.2.3 Mức vật lý

```

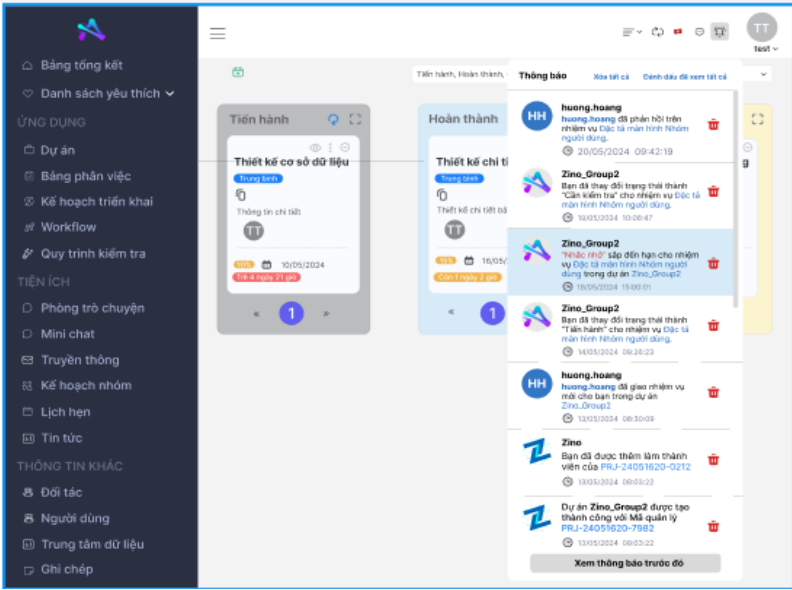
DROP TABLE IF EXISTS `ta_msg_message`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `ta_msg_message` (
  `I_ID` int(11) NOT NULL AUTO_INCREMENT,
  `I_Status` int(11) NOT NULL COMMENT 'mac dinh trong chuong trinh',
  `I_Type_01` int(11) NOT NULL COMMENT 'type Msg: 1:email, 2:chat',
  `I_Type_02` int(11) DEFAULT NULL COMMENT 'type Noti: envoi sms, envoi email, in-app',
  `T_Info_01` varchar(2000) DEFAULT NULL COMMENT 'T_From',
  `T_Info_02` varchar(2000) DEFAULT NULL COMMENT 'T_To',
  `T_Info_03` varchar(2000) DEFAULT NULL COMMENT 'T_Title',
  `T_Info_04` longtext DEFAULT NULL COMMENT 'T_Body',
  `T_Info_05` longtext DEFAULT NULL,
  `I_Aut_User` int(11) DEFAULT NULL COMMENT 'user creates',
  `D_Date_01` datetime DEFAULT NULL,
  `D_Date_02` datetime DEFAULT NULL,
  `I_Entity_Type` int(11) DEFAULT NULL,
  `I_Entity_ID` int(11) DEFAULT NULL,
  PRIMARY KEY (`I_ID`),
  KEY `idx_TMMES_01` (`I_Aut_User`),
  KEY `idx_TMMES_02` (`I_Entity_Type`,`I_Entity_ID`),
  --
  PRIMARY KEY (`I_ID`),
  KEY `idx_TMMES_01` (`I_Aut_User`),
  KEY `idx_TMMES_02` (`I_Entity_Type`,`I_Entity_ID`),
  KEY `idx_TMMES_03` (`D_Date_01`,`D_Date_02`)
) ENGINE=InnoDB AUTO_INCREMENT=2319 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

```

Hình 4.6. CSDL mức vật lý

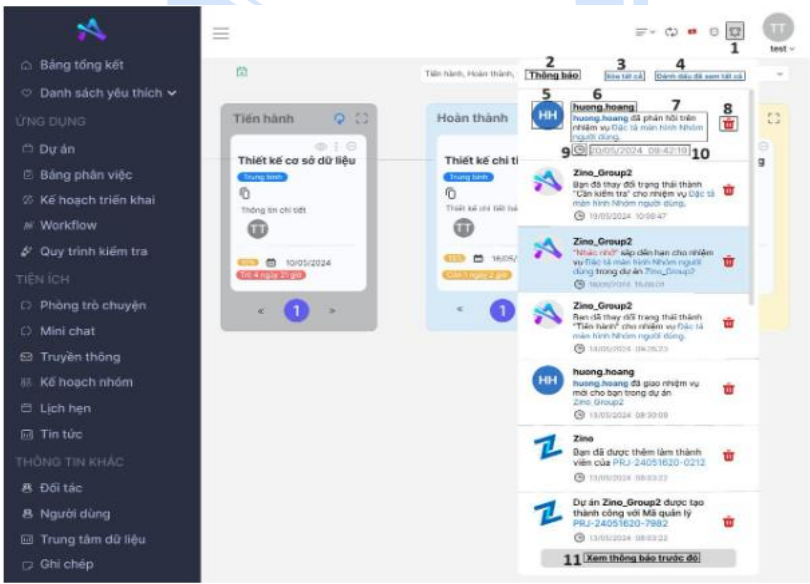
### 4.3. Thiết kế giao diện và đặc tả giao diện.

4.3.1 Thiết kế giao diện.




Hình 4.7: Giao diện thông báo.



4.3.2 Đặc tả giao diện.



Hình 4.8. Đặc tả giao diện thông báo

STT	Loại	Mô tả	Ghi chú
1	Button	<ul style="list-style-type: none"><li>Hiện thị cứng: </li><li>Sự kiện: khi nhấn vào thì hiển thị màn hình những thông báo</li></ul>	



2	Text	<ul style="list-style-type: none"> <li>• Hiện thị cứng: <b>‘Thông báo’</b></li> </ul>	
3	Button	<ul style="list-style-type: none"> <li>• Hiện thị cứng text : <b>‘xóa tất cả’</b></li> <li>• Sự kiện: Khi nhấn vào thì sẽ xóa tất cả những thông báo</li> </ul>	
4	Button	<ul style="list-style-type: none"> <li>• Hiện thị cứng text : <b>‘đánh dấu đã xem tất cả’</b></li> <li>• Sự kiện: Khi nhấn vào thì những thông báo có trạng thái đã xem</li> </ul>	
5	Text	<ul style="list-style-type: none"> <li>• Hiện thị cứng Tên viết tắt của user</li> </ul>	
6	Text	<ul style="list-style-type: none"> <li>• Hiện thị tên đầy đủ của user</li> </ul>	
7	Text	<ul style="list-style-type: none"> <li>• Hiện thị tên user và chức năng của user khi tác động vào 1 nhiệm vụ nào đó</li> </ul>	
8	Button	<ul style="list-style-type: none"> <li>• Hiện thị cứng: </li> <li>• Sự kiện: khi nhấn vào thì sẽ xóa thông báo đó</li> </ul>	
9	Icon	<ul style="list-style-type: none"> <li>• Hiện thị cứng: </li> </ul>	
10	Icon	<ul style="list-style-type: none"> <li>• Hiện thị cứng thời gian mà user tác động vào 1 chức năng</li> </ul>	
11	Text	<ul style="list-style-type: none"> <li>• Hiện thị cứng: <b>‘xem thông báo trước đó’</b></li> <li>• Sự kiện: Khi nhấn vào thì hệ thống sẽ hiển thị những thông báo trước đó</li> </ul>	

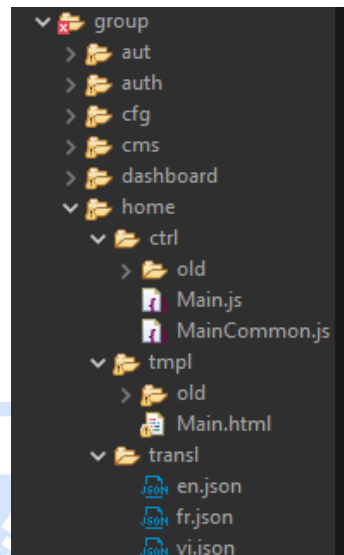
Bảng 4.7: Đặc tả giao diện thông báo

## CHƯƠNG 5. PHÁT TRIỂN ỨNG DỤNG.

### 5.1. Phát triển front-end.

#### 5.1.1 Cấu trúc Framework.

Sử dụng Framework của công ty INOTEV



Hình 5.1. Hình Cấu trúc Framework

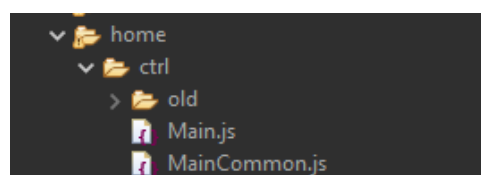
Mỗi chức năng đều có ba thư mục chính: **controller**, **template**, và **translate**:

- **Controller:** Chứa các tệp JavaScript của chức năng.
- **Template:** Chứa các tệp HTML của chức năng.
- **Translate:** Chứa các tệp JSON dùng để dịch thuật ngôn ngữ. Hiện tại, hệ thống hỗ trợ ba ngôn ngữ: Tiếng Anh, Tiếng Pháp và Tiếng Việt.

#### 5.1.2 Thiết kế các Controller

Cấu trúc của Controller gồm có file main và các file phụ tương ứng với các chức năng.

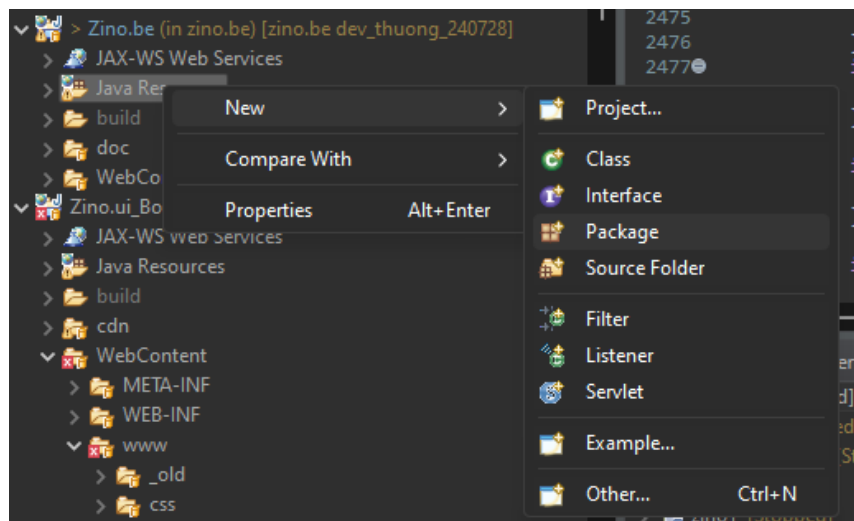
Chức năng thông báo có file như sau:



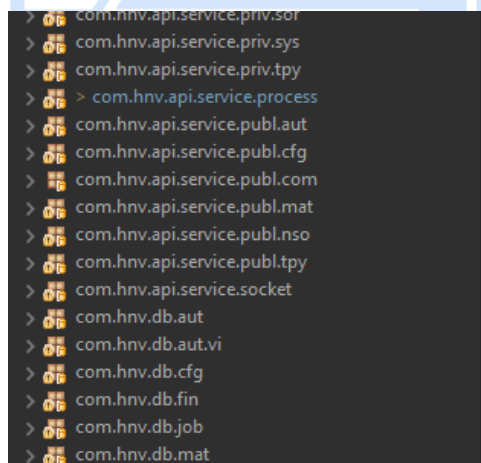
Hình 5.2. Hình Cấu trúc của controller

## 5.2. Phát triển back-end.

### 5.2.1 Khởi tạo package



Hình 5.3. Cách khởi tạo package



Hình 5.4. Sau khi khởi tạo

### 5.2.2 API.

#### a. API chức năng thông báo.

Service	Ý nghĩa	Tham số
ServiceMsgMessage	Quản lý các thông báo	Không có
ServicePrjProject	Gọi thông báo liên quan tới dự án	Không có
ServiceNsoGroup	Gọi thông báo liên quan tới lịch hẹn	Không có

ServiceNsoPost	Gọi thông báo liên quan tới tin tức	Không có
ProcessMsg	Tạo các thông báo.	Không có
ProcessMain	Gọi các thông báo tự động	Không có

Bảng 5.1. API sử dụng trong chức năng thông báo

**b. Triển khai API cho các thông báo.**

**Thông báo khi thêm mới dự án/ phân việc:**

- Thông báo này sẽ gửi tới tất cả các thành viên được thêm vào dự án hoặc phân việc ngoại trừ người tạo dự án, phân việc.
- API sử dụng: ProcessMsg và ServicePrjProject

```
// Thêm vào thêm project/ task mới, comment, chuẩn task, thay đổi thông tin
public static void reqsaveNotificationComment(TaPrjProject prj, TaAutUser user, String table_notif, int typ_notif) throws Exception {
    Criterion cri = Restrictions.and(
        Restrictions.eq(TaPrjRelationship.ATT_I_ENTITY_TYPE_01, DefDBExt.ID_TA_PRJ_PROJECT),
        Restrictions.eq(TaPrjRelationship.ATT_I_ENTITY_TYPE_02, DefDBExt.ID_TA_AUT_USER),
        Restrictions.eq(TaPrjRelationship.ATT_I_ENTITY_ID_01, prj.reqId()),
        Restrictions.ne(TaPrjRelationship.ATT_I_ENTITY_ID_02, user.reqId())
    );
    List<TaPrjRelationship> lstMem = TaPrjRelationship.DAO.reqList(cri);
    Set<Integer> setMem = new HashSet<Integer>();
    if (lstMem != null && lstMem.size() > 0) {
        setMem = ToolSet.reqSetInt(lstMem, TaPrjRelationship.ATT_I_ENTITY_ID_02);
        if (setMem != null && setMem.size() > 0) {
            setMem.remove(user.reqId()); //---no need noti for actual user
        }
    }
    //---build content of noti
    JSONArray arr = new JSONArray();
    for (Integer mem: setMem) {
        JSONObject notify = new JSONObject();
        notify.put("typ", typ_notif);
        notify.put("typTab", table_notif);
        notify.put("title", prj.req(TaPrjProject.ATT_I_NAME));
        notify.put("code", prj.req(TaPrjProject.ATT_I_CODE_01));
        notify.put("stat", prj.req(TaPrjProject.ATT_I_STATUS_01));
        notify.put("login", user.req(TaAutUser.ATT_I_LOGIN_01));
        notify.put("typ82", prj.req(TaPrjProject.ATT_I_TYPE_02));
        notify.put("uid", mem);
        notify.put("uAction", user.reqRef());
        notify.put("parTyp", DefDBExt.ID_TA_PRJ_PROJECT);
        notify.put("parID", prj.reqId());
        arr.add(notify);
    }
    //---insert into db
    reqNewNoti(prj.reqId(), arr);
}
```

- Dưới đây là dòng lệnh để gọi thông báo trong hàm reqnew ở ServicePrjProject

```
//req to Noti
ProcessMsg.reqSaveNotificationComment(ent, user, TAB_PRJ, TYP_ADD);
```

STT	Tên	Dữ liệu được thêm vào từ	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_ADD	Loại thông báo: thêm mới	<pre>{   "parTyp": 250000,   "uID": 28,   "stat": 100200,   "code": "PRJ-24072909-3019",   "parID": 1344,   "typTab": "prj",   "typ02": 0,   "uAction": 1,   "typ": 1,   "title": "dự án test",   "login": "adm" }</pre>
2	typTab	TAB_PRJ	Thông báo khi thêm mới prj/task	
3	title	TaPrjProject.ATT_T_NAME	Tên của dự án	
4	code	TaPrjProject.ATT_T_CODE_01	Mã code của dự án	
5	stat	TaPrjProject.ATT_I_STAT_US_01	Trạng thái của dự án	
6	login	TaAutUser.ATT_T_LOGIN_01	Lưu lại lịch sử đăng nhập	
7	uID	List<TaTpyRelationship> IstMem = TaTpyRelationship.DAO.reqList(crl);	Lưu lại ID của từng thành viên trong list danh sách thành viên.	
8	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
9	parTyp	DefDBExt.ID_TA_PRJ_PROJECT	Loại đối tượng của Project	
10	parID	id của prj	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.5. Dữ liệu json nhận được của thông báo thêm dự án/ phân việc.

#### Thông báo thêm mới comment vào dự án/ phân việc:

- Thông báo này sẽ gửi thông báo cho tất cả các thành viên có trong dự án và ngoại trừ thành viên thực hiện hành động comment.
- API sử dụng: ProcessMsg và ServicePrjProject.

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm dosavecomment() ở API ServicePrjProject:

```
ProcessMsg.reqSaveNotificationComment(ent, user, TAB_COMMENT, TYP_ADD);
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_ADD	Loại thông báo: thêm mới	<pre>{   "parTyp": 250000,   "uID": 27,   "stat": 100200,   "code": "PRJ-24072909-3019",   "parID": 1344,   "typTab": "comment",   "comment": "typ02": 0,   "uAction": 1,   "typ": 1,   "title": "dự án test",   "login": "adm" }</pre>
2	typTab	TAB_COMMENT	Thông báo khi thêm mới bình luận vào project/task	
3	title	TaPrjProject.ATT_T_NAM E	Tên của dự án	
4	code	TaPrjProject.ATT_T_COD E_01	Mã code của dự án	
5	stat	TaPrjProject.ATT_I_STAT US_01	Trạng thái của dự án	
6	login	TaAutUser.ATT_T_LOGI N_01	Lưu lại lịch sử đăng nhập	
7	uID	ID của từng thành viên trong list danh sách thành viên List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList(crl);	Lưu lại ID của từng thành viên trong list danh sách thành viên.	
8	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
9	parTyp	DefDBExt.ID_TA_PRJ_P ROJECT	Loại đối tượng của Project	
10	parID	id của prj	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.6. Dữ liệu json nhận được của thông báo thêm comment

### Thông báo khi thay đổi thông tin dự án

- Khi có thay đổi các thông tin như dưới ngoại trừ trạng thái thì gửi thông báo tới các thành viên trong dự án ngoại trừ thành viên đã thực hiện thay đổi.

Mã quản lý PRJ-24051620-7982	Tag #dacta	Lĩnh vực Tin học	Cấp độ Trung bình
Trạng thái Cần làm	Ngân sách	Giá trị thực tế	% hoàn thành
KPI %	SLA (phút)	Ngày bắt đầu 31/08/2024 08:00:00	Ngày kết thúc 30/09/2024 18:00:00

- API sử dụng : Process và ServicePrjProject.

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm reqSaveContent() ở API ServicePrjProject :

```
ProcessMsg.reqSaveNotificationComment(ent, user, TAB_CONTENT, TYP_MOD);
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_MOD	Loại thông báo: sửa đổi	<pre>{   "parTyp": 250000,   "ulID": 27,   "stat": 100200,   "code": "PRJ-24071823-9938",   "parID": 1334,   "typTab": "content",   "typ02": 0,   "uAction": 1,   "typ": 2,   "title": "dự án mới",   "login": "adm" }</pre>
2	typTab	TAB_CONTENT	Thông báo khi thay đổi thông tin của dự án.	
3	title	TaPrjProject.ATT_T_NAM E	Tên của dự án	
4	code	TaPrjProject.ATT_T_COD E_01	Mã code của dự án	
5	stat	TaPrjProject.ATT_I_STAT US_01	Trạng thái của dự án	
6	login	TaAutUser.ATT_T_LOGI N_01	Lưu lại lịch sử đăng nhập	
7	ulID	ID của từng thành viên trong list danh sách thành viên List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList(crl);	Lưu lại ID của từng thành viên trong list danh sách thành viên.	
8	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
9	parTyp	DefDBExt.ID_TA_PRJ_P ROJECT	Loại đối tượng của Project	
10	parID	id của prj	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.7 Dữ liệu json nhận được của thông báo sửa thông tin dự án

### Thông báo khi chuyển phân việc.

- Khi một phân việc được thay đổi trạng thái (vd từ khởi tạo -> cần làm) thì sẽ gửi thông báo tới các thành viên có trong phân việc trừ thành viên thực hiện chuyển đổi phân việc.
- API sử dụng : Process và ServicePrjProject

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm reqSaveContent() ở API ServicePrjProject:

```
if ( mapChange.containsKey(TaPrjProject.ATT_I_STATUS_01) && (
    statNew == TaPrjProject.STAT_01_PRJ_TODO || statNew == TaPrjProject.STAT_01_PRJ_INPROGRESS
    || statNew == TaPrjProject.STAT_01_PRJ_REVIEW || statNew == TaPrjProject.STAT_01_PRJ_TEST ) ){
    ProcessMsg.reqSaveNotificationComment(ent, user, TAB_CONTENT, TYP_MOVE);

    //add logic for change stat of other workflow to inactive when this workflow is active
}
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_MOVE	Loại thông báo: di chuyển	<pre>{   "parTyp": 250000,   "uID": 28,   "stat": 100300,   "code": "PRJ-24072421-2",   "parID": 1336,   "typTab": "content",   "content": "typ02": 2,   "uAction": 1,   "typ": 9,   "title": "phân việc 02",   "login": "adm" }</pre>
2	typTab	TAB_CONTENT	Thông báo khi chuyển task	
3	title	TaPrjProject.ATT_T_NAM E	Tên của dự án	
4	code	TaPrjProject.ATT_T_COD E_01	Mã code của dự án	
5	stat	TaPrjProject.ATT_I_STAT US_01	Trạng thái của dự án	
6	login	TaAutUser.ATT_T_LOGI N_01	Lưu lại lịch sử đăng nhập	
7	uID	ID của từng thành viên trong list danh sách thành viên List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList(crl);	Lưu lại ID của từng thành viên trong list danh sách thành viên.	
8	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
9	parTyp	DefDBExt.ID_TA_PRJ_P ROJECT	Loại đối tượng của Project	
10	parID	id của prj	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.8. Dữ liệu json nhận được của thông báo chuyển phân việc

### Thông báo thêm thành viên vào phân việc hoặc dự án:

- Thông báo này sẽ gửi thông báo cho tất cả các thành viên được thêm vào dự án hoặc phân việc.
- API sử dụng : Process và ServicePrjProject.

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm doSaveMember() ở API ServicePrjProject:

```
// thông báo thêm mới thành viên vào prj/task
public static void doBuildNotiArray(List<TaTpyRelationship> lst, JSONArray arr, TaAutUser user, TaPrjProject prj, Integer typMember, Integer typAction) {
    Set<Integer> setMem = new HashSet<Integer>();

    if (lst != null && lst.size() > 0) {
        setMem.addAll(ToolSet.reqSetInt(lst, TaTpyRelationship.ATT_I_ENTITY_ID_02));
    }

    if (!setMem.isEmpty()) {
        for (Integer mem: setMem) {
            JSONObject notify = reqNotiJson(user, prj, typMember, typAction, mem);
            arr.add(notify);
        }
    }
}

public static JSONObject reqNotiJson(TaAutUser user, TaPrjProject prj, Integer typMember, Integer typAction, Integer mem) {
    JSONObject notify = new JSONObject();
    notify.put("typ", typMember);
    notify.put("typTab", TAB_MEMBER);
    notify.put("title", prj.req(TaPrjProject.ATT_T_NAME));
    notify.put("code", prj.req(TaPrjProject.ATT_T_CODE_01));
    notify.put("id", prj.req(TaPrjProject.ATT_T_ID));
    notify.put("typ02", prj.req(TaPrjProject.ATT_I_TYPE_02));
    notify.put("uID", user.req());
    notify.put("uAction", user.reqRef());
    notify.put("parTyp", DefDBExt.ID_TA_PRJ_PROJECT);
    notify.put("parID", prj.reqId());

    return notify;
}
```



```

Integer stat = (Integer) ent.req(TaPrjProject.ATT_I_STATUS_01);
if (stat == TaPrjProject.STAT_01_PRJ_TODO || stat == TaPrjProject.STAT_01_PRJ_INPROGRESS || stat == TaPrjProject.STAT_01_PRJ_REVIEW || stat == TaPrjProject.STAT_01_PRJ_TEST)
{
    //Gửi thông báo
    JSONArray arrNotify = new JSONArray();
    //
    ProcessMsg.doBuildNotify(list, arrNotify, user, ent, TYP_JOIN, TYP_ADD);
    ProcessMsg.regNewMoti(ent.reqId(), arrNotify);
}

```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_ADD	Loại thông báo: thêm mới	<pre> {   "parTyp": 250000,   "ulID": 1,   "parID": 1334,   "typTab": "member",   "typ02": 0,   "uAction": 1,   "typ": 4,   "id": 1334,   "title": "dự án mới",   "code": "PRJ-24071823-9938" } </pre>
2	typTab	TAB_MEMBER	Thông báo khi thêm mới thành viên vào prj/ task	
3	title	TaPrjProject.ATT_I_NAME	Tên của dự án	
4	code	TaPrjProject.ATT_I_CODE_01	Mã code của dự án	
5	id	TaPrjProject.ATT_I_ID	Trạng thái của dự án	
6	typ02	TaPrjProject.ATT_I_TYP_02	Lưu lại lịch sử đăng nhập	
7	ulID	ID của từng thành viên trong list danh sách thành viên List<TaTypRelationship> lstMem = TaTypRelationship.DAO.reqList(crit);	Lưu lại ID của từng thành viên trong list danh sách thành viên.	
8	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
9	parTyp	DefDBExt.ID_TA_PRJ_PROJECT	Loại đối tượng của Project	
10	parID	id của prj	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.9. Dữ liệu json nhận được của thông báo thêm thành viên.

### Thông báo thêm mới lịch hẹn:

- Thông báo này sẽ gửi thông báo cho tất cả các thành viên được thêm vào lịch hẹn
- API sử dụng: Process và ServiceNsoGroup

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm doNew() và doMod() ở API ServiceNsoGroup:

```

// thông báo thêm mới lịch hẹn.
public static void sendNotificationNewGroup(TaAutUser user, TaNsoGroup ent) throws Exception {
    List<TaNsoGroupMember> members = TaNsoGroupMember.DAO.reqList(
        Restrictions.eq(TaNsoGroupMember.COL_I_NSO_GROUP, ent.reqId())
    );

    JSONArray arrNotify = new JSONArray();

    if (members != null) {
        for (TaNsoGroupMember member : members) {
            JSONObject notify = new JSONObject();
            notify.put("typTab", TAB_NSO);
            notify.put("typ", TYP_ADD);
            notify.put("title", ent.req(TaNsoGroup.ATT_I_NAME));
            notify.put("code", ent.req(TaNsoGroup.ATT_I_INFO_01));
            notify.put("ulID", member.req(TaNsoGroupMember.ATT_I_AUT_USER));
            notify.put("uAction", user.reqRef());
            notify.put("parTyp", DefDBExt.ID_TA_NSO_GROUP);
            arrNotify.add(notify);
        }
    }
    regNewNoti(ent.reqId(), arrNotify);
}

```

```
user.reqId(), user.reqStr(TaAutUser.ATT_T_LO
ProcessMsg.sendNotificationNewGroup(user, ent);
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_ADD	Loại thông báo: thêm mới	{ "parTyp": 5000, "uID": 28, "typTab": "appointment", "uAction": 1, "typ": 1, "title": "Họp đầu tuần" }
2	typTab	TAB_NSO	Thông báo khi thêm mới lịch hẹn.	
3	title	TaNsoGroup.ATT_T_NA ME	Tiêu đề lịch hẹn	
4	uID	TaNsoGroupMember.ATT _I_AUT_USER	ID của người tạo lịch hẹn.	
5	uAction	user.reqRef()	Lưu lại dùng để tham chiếu user.	
6	parTyp	DefDBExt.ID_TA_NSO_ GROUP	ID của đối tượng TaNsoGroup Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.10. Dữ liệu json nhận được của thông báo thêm mới lịch hẹn.

### Thông báo thêm mới bài viết:

- Thông báo này sẽ chỉ gửi thông báo cho quản trị viên người thực hiện duyệt bài viết.
- API sử dụng: Process và ServiceNsoPost.

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm doNewNews() ở API ServiceNsoGroup:

```
// Thông báo thêm mới bài viết, duyệt bài viết
public static void sendNotificationPost(TaAutUser user, TaNsoPost ent, int typ) throws Exception {
    JSONArray arrNotify = new JSONArray();
    JSONObject notify = new JSONObject();

    notify.put("typTab", TAB_POST);
    notify.put("typ", typ);
    notify.put("title", ent.req(TaNsoPost.ATT_T_TITLE));
    notify.put("code", ent.req(TaNsoPost.ATT_T_CODE_01));
    notify.put("stat", ent.req(TaNsoPost.ATT_T_STATUS_01));
    notify.put("uID", user.reqId());
    notify.put("uAction", user.reqRef());
    notify.put("parTyp", DefDBExt.ID_TA_NSO_POST);
    arrNotify.add(notify);

    reqNewNoti(ent.reqId(), arrNotify);
}
```

```
TaAutUser autUser = TaAutUser.DAO.reqEntityByKey(mail
ProcessMsg.sendNotificationPost(autUser, ent, TYP_NEWS);
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_NEWS	loại thông báo: loại tin tức.	<pre>{   "parTyp": 200001,   "uID": 1,   "stat": 0,   "code": "O_2407290957",   "typTab": "post",   "uAction": 1,   "typ": 7,   "title": "bài viết mới" }</pre>
2	typTab	TAB_POST	Thông báo khi thêm mới bài viết.	
3	title	TaNsoPost.ATT_T_TITLE	Tiêu đề bài viết	
4	code	(TaNsoPost.ATT_T_CODE_01)	mã code bài viết	
5	stat	TaNsoPost.ATT_I_STATU S_01	Trạng thái của bài viết	
6	uID	user.reqId()	ID của người dùng	
7	uAction	user.reqRef()	Tham chiếu hành động của người dùng	
8	parTyp	DefDBExt.ID_TA_NSO_GROUP	ở đây là ID của đối tượng nhóm (TaNsoGroup). Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.11. Dữ liệu json nhận được của thông báo thêm mới tin tức.

#### Thông báo khi bài viết được duyệt thành công:

- Thông báo này sẽ gửi thông báo tới người gửi bài đăng sau khi được admin duyệt.
- API sử dụng: Process và ServiceNsoPost.

Phần thông báo này sẽ kế thừa lại hàm ở ProcessMsg và được gọi trong hàm reqMod() ở API ServiceNsoGroup:

```
ProcessMsg.sendNotificationPost(autoUser, ent, TYP_APPROVAL);
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_APPROVAL	loại thông báo: loại duyệt bài viết	<pre>{   "parTyp": 200001,   "uID": 1,   "stat": 2,   "code": "O_2407290957",   "typTab": "post",   "uAction": 1,   "typ": 8,   "title": "bài viết mới" }</pre>
2	typTab	TAB_POST	Thông báo khi bài viết được duyệt hoặc không được duyệt.	
3	title	TaNsoPost.ATT_T_TITLE	Tiêu đề bài viết	
4	code	(TaNsoPost.ATT_T_CODE_01)	mã code bài viết	
5	stat	TaNsoPost.ATT_I_STATU S_01	Trạng thái của bài viết	
6	uID	user.reqId()	ID của người dùng	
7	uAction	user.reqRef()	Tham chiếu hành động của người dùng	
8	parTyp	DefDBExt.ID_TA_NSO_GROUP	ở đây là ID của đối tượng nhóm (TaNsoGroup). Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.12. Dữ liệu json nhận được của thông báo bài viết được duyệt.

## Thông báo phân việc bị trễ hạn:

- Thông báo này sẽ gửi tới các thành viên của phân việc trễ đó. Thông báo này chỉ được gửi trong 3 ngày, quá thời gian đó sẽ không gửi thông báo nữa.

- API sử dụng : ProcessMsg, ProcessMain

⇒ Thông báo này sẽ được kế thừa từ ProcessMsg và được gọi ở ProcessMain

```
// Thông báo nhiệm vụ quá hạn
public static void do_CheckTaskExpired() {
    Thread t = new Thread() {
        public void run() {
            try {
                Date dtNow = new Date();
                Date dtLim = ToolDate.reqDateByAdding(dtNow, 0, 0, -3, 0, 0, 0);

                // Nhiệm vụ quá hạn (dtLim < ATT_D_DATE_04 < dtNow)
                // Chỉ nhắc account trong 3 ngày
                List<TaPrjProject> overdueTasks = TaPrjProject.DAO.reqList(
                    Restrictions.in(TaPrjProject.ATT_I_STATUS_01, statForCheck),
                    Restrictions.between(TaPrjProject.ATT_D_DATE_04, dtLim, dtNow)
                );

                Hashtable<Integer, TaPrjProject> tabTask = new Hashtable<>();
                for (TaPrjProject task : overdueTasks) {
                    tabTask.put(task.reqId(), task);
                }

                if (!tabTask.isEmpty()) {
                    List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList_In(TaTpyRelationship.ATT_I_ENTITY_ID_01, tabTask.keySet(),
                        Restrictions.eq(TaTpyRelationship.ATT_I_ENTITY_TYPE_01, DefDBExt.ID_TA_PRJ_PROJECT));
                    reqSaveNotificationTask(overdueTasks.get(0).reqId(), TYP_LATE, lstMem, tabTask);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    };
    ThreadManager.doExecuteInfini(t, DefTime.TIME_00_05_00_000, DefTime.TIME_00_05_00_000);
}
```

```
private static void reqSaveNotificationTask(int prjId, int type, List<TaTpyRelationship> lstMem, Hashtable<Integer, TaPrjProject> tabTask) throws Exception {
    JSONArray arr = new JSONArray();
    if (lstMem != null && lstMem.size() > 0) {
        for (TaTpyRelationship mem : lstMem) {
            TaPrjProject prj = tabTask.get(mem.req(TaTpyRelationship.ATT_I_ENTITY_ID_01));

            JSONObject notify = new JSONObject();
            notify.put("tblTab", TAB_TASK);
            notify.put("typ", type);
            notify.put("title", prj.req(TaPrjProject.ATT_I_NAME));
            notify.put("code", prj.req(TaPrjProject.ATT_I_CODE_01));
            notify.put("code", prj.req(TaPrjProject.ATT_I_CODE_01));
            notify.put("parTyp", DefDBExt.ID_TA_PRJ_PROJECT);
            notify.put("parID", prj.reqId());
            arr.add(notify);
        }
    }
    reqNewNoti(prjId, arr);
}
```

```
//---create noti for task expired/expire soon and meetings
ProcessMsg.do_CheckTaskExpired();
ProcessMsg.do_CheckTaskSoon();
ProcessMsg.do_CheckMeetingSoon();
ProcessMsg.do_CheckPeriodicReminders();
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_LATE	Loại thông báo: trễ	{ "parTyp": 250000, "uID": 28, "code": "PRJ-240724 21-2048", "parID": 1336, "typTab": "task", "title": "test01", "typ": 10 }
2	typTab	TAB_TASK	Thông báo khi task bị trễ	
3	title	TaPrjProject.ATT_T_NAME	Tiêu đề dự án	
4	code	TaPrjProject.ATT_T_CODE _01	mã code dự án	
7	uID	TaTpyRelationship.ATT_I_ ENTITY_ID_02	ID của người dùng liên quan	
9	parTyp	DefDBExt.ID_TA_PRJ_PR OJECT	Loại đối tượng của Project	
10	parID	prj.reqId()	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.13 Dữ liệu json nhận được của thông báo phân việc sắp trễ hạn.

### Thông báo phân việc sắp trễ hạn:

- Thông báo này sẽ gửi tới các thành viên của phân việc sắp trễ hạn đó. Thông báo này sẽ được gửi trước hạn là 12 tiếng.

- ProcessMsg, ProcessMain.

⇒ Thông báo này sẽ được kế thừa từ ProcessMsg và được gọi ở ProcessMain.

```
// Thông báo nhiệm vụ sắp trễ hạn
public static void do_checkTaskSoon() {
    Thread t = new Thread() {
        public void run() {
            try {
                Date dtNow = new Date();
                Date dtLim = ToolDate.reqDateByAdding(dtNow, 0, 0, 12, 0, 0);

                // Nhiệm vụ sắp đến hạn (dtNow < ATT_D_DATE_04 < dtLim)
                List<TaPrjProject> overdueTasks = TaPrjProject.DAO.reqList(
                    Restrictions.in (TaPrjProject.ATT_T_STATUS_01, statForCheck),
                    Restrictions.between(TaPrjProject.ATT_D_DATE_04, dtNow, dtLim)
                );

                Hashtable<Integer, TaPrjProject> tabTask = new Hashtable<>();
                for (TaPrjProject task : overdueTasks) {
                    tabTask.put(task.reqId(), task);
                }

                if (tabTask.isEmpty()) {
                    List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList_In(TaTpyRelationship.ATT_I_ENTITY_ID_01, tabTask.keySet(),
                        Restrictions.eq(TaTpyRelationship.ATT_I_ENTITY_TYPE_01, DefDBExt.ID_TA_PRJ_PROJECT));
                    reqSaveNotificationTask(overdueTasks.get(0).reqId(), TYP_SOON, lstMem, tabTask);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    };
    ThreadManager.doExecuteInfini(t, DefTime.TIME_00_05_00_000, DefTime.TIME_00_05_00_000);
}

private static void reqSaveNotificationTask(int prjId, int type, List<TaTpyRelationship> lstMem, Hashtable<Integer, TaPrjProject> tabTask) throws Exception {
    JSONArray arr = new JSONArray();
    if (lstMem != null && lstMem.size() > 0) {
        for (TaTpyRelationship mem : lstMem) {
            TaPrjProject prj = tabTask.get(mem.req(TaTpyRelationship.ATT_I_ENTITY_ID_01));

            JSONObject notify = new JSONObject();
            notify.put("typTab", TAB_TASK);
            notify.put("type", type);
            notify.put("title", prj.req(TaPrjProject.ATT_T_NAME));
            notify.put("code", prj.req(TaPrjProject.ATT_T_CODE_01));
            notify.put("uID", mem.req(TaTpyRelationship.ATT_I_ENTITY_ID_02));
            notify.put("parTyp", DefDBExt.ID_TA_PRJ_PROJECT);
            notify.put("parID", prj.reqId());
            arr.add(notify);
        }
    }
    reqNewNoti(prjId, arr);
}

//---create noti for task expired/expire soon and meetings
ProcessMsg.do_CheckTaskExpired();
ProcessMsg.do_CheckTaskSoon();
ProcessMsg.do_CheckMeetingSoon();
ProcessMsg.do_CheckPeriodicReminders();|
```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_SOON	Loại thông báo: Gần tới hạn	{ "parTyp": 250000, "uID": 28, "code": "PRJ-2407 2421-8261 ", "parID": 1335, "typTab": "task", "typ":5 "title": "phân việc 1" }
2	typTab	TAB_TASK	Thông báo khi task sắp gần tới hạn	
3	title	TaPrjProject.ATT_T_NAME	Tiêu đề dự án	
4	code	TaPrjProject.ATT_T_CODE_01	mã code dự án	
7	uID	TaTpyRelationship.ATT_I_ENTITY_ID_02	ID của người dùng liên quan	
9	parTyp	DefDBExt.ID_TA_PRJ_PROJECT	Loại đối tượng của Project	
10	parID	prj.reqId()	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.14. Dữ liệu json nhận được của thông báo phân việc sắp trễ hạn.

### Thông báo định kỳ hằng tuần

- Thông báo này sẽ gửi thông báo cho thành viên ở trong prj số phân việc chưa hoàn thành. sẽ gửi 1 tuần 1 lần.
  - API sử dụng : ProcessMsg, ProcessMain.
- ⇒ Thông báo này sẽ được kế thừa từ ProcessMsg và được gọi ở ProcessMain.

```

public static void do_CheckPeriodicReminders() {
    Thread t = new Thread() {
        public void run() {
            try {
                Date currentDate = new Date();
                // Tính đến cuối tuần (tính tuần mới lần)
                Date reminderLimit = ToolDate.reqDateByAdding(currentDate, 0, 0, -7, 0, 0, 0);

                List<TaPrjProject> projects = TaPrjProject.DAO.reqList(
                    Restrictions.gt(TaPrjProject.ATT_D_DATE_04, reminderLimit),
                    Restrictions.lt(TaPrjProject.ATT_D_DATE_04, currentDate)
                );

                Hashtable<Integer, TaPrjProject> tabTask = new Hashtable<>();

                for (TaPrjProject project : projects) {
                    List<TaPrjProject> tasks = TaPrjProject.DAO.reqList(
                        Restrictions.eq(TaPrjProject.ATT_I_ID, project.reqId()),
                        Restrictions.gt(TaPrjProject.ATT_D_DATE_04, reminderLimit),
                        Restrictions.lt(TaPrjProject.ATT_D_DATE_04, currentDate)
                    );

                    for (TaPrjProject task : tasks) {
                        tabTask.put(task.reqId(), task);
                    }
                }

                if (tabTask.isEmpty()) {
                    List<TaTpyRelationship> lstMem = TaTpyRelationship.DAO.reqList_In(TaTpyRelationship.ATT_I_ENTITY_ID_01, tabTask.keySet(),
                        Restrictions.eq(TaTpyRelationship.ATT_I_ENTITY_TYPE_01, DefDBExt.ID_TA_PRJ_PROJECT)
                    );
                    regSavePeriodicReminders(lstMem, tabTask);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    };
}

```

```

private static void reqSavePeriodicReminders(List<TaTpyRelationship> lstMem, Hashtable<Integer, TaPrjProject> tabTask) throws Exception {
    JSONArray arr = new JSONArray();
    int countStat = 0;
    Set<Integer> setIdsPrj = tabTask.keySet();

    Date currentDate = new Date();
    SimpleDateFormat monthFormat = new SimpleDateFormat("MM");
    String currentMonth = monthFormat.format(currentDate);

    for (Integer key : setIdsPrj) {
        TaPrjProject prj = tabTask.get(key);
        if (statForCheck.contains(prj.req(TaPrjProject.ATT_I_STATUS_01))) {
            countStat++;
        }
    }

    if (lstMem != null && !lstMem.isEmpty()) {
        Set<Integer> notifiedUserIds = new HashSet<>();

        for (TaTpyRelationship mem : lstMem) {
            int userId = (int) mem.req(TaTpyRelationship.ATT_I_ENTITY_ID_02);
            TaPrjProject prj = tabTask.get(mem.req(TaTpyRelationship.ATT_I_ENTITY_ID_01));
            if (!notifiedUserIds.contains(userId)) {
                JSONObject notify = new JSONObject();
                notify.put("typTab", TAB_REMINDER);
                notify.put("typ", TYP_REMINDER);
                notify.put("count", countStat);
                notify.put("month", currentMonth);
                notify.put("parTyp", DefDBExt.ID_TA_PRJ_PROJECT);
                notify.put("ulID", userId);
                notify.put("parID", prj.reqId());
                arr.add(notify);
                notifiedUserIds.add(userId);
            }
        }
    }
}

```

```

//---create noti for task expired/expire soon and meetings
ProcessMsg.do_CheckTaskExpired();
ProcessMsg.do_CheckTaskSoon();
ProcessMsg.do_CheckMeetingSoon();
ProcessMsg.do_CheckPeriodicReminders();

```

STT	Tên	Dữ liệu được thêm vào	Ý nghĩa	Dữ liệu từ json
1	typ	TYP_REMINDER	Loại thông báo: Định kỳ	<pre> {   "parTyp": 250000,   "ulID": 1,   "month": "07",   "parID": 1332,   "typTab": "reminder",   "count": 3,   "typ": 12 } </pre>
2	typTab	TAB_REMINDER	Thông báo định kỳ	
3	count	<pre> for (Integer key : setIdsPrj) {   TaPrjProject prj = tabTask.get(key);   if (statForCheck.contains(prj.req(TaPrjProject.ATT_I_STATUS_01))) {     countStat++;   } } </pre>	đếm số lượng các task chưa hoàn thành.	
4	month	<pre> Date currentDate = new Date(); SimpleDateFormat monthFormat = new SimpleDateFormat("MM"); String currentMonth = monthFormat.format(currentDate); </pre>	Tháng hiện tại khi nhận thông báo	
5	ulID	TaTpyRelationship.ATT_I_ENTITY_ID_02	ID của người dùng liên quan	
6	parTyp	DefDBExt.ID_TA_PRJ_PROJECT	Loại đối tượng của Project	
7	parID	prj.reqId()	id của prj Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.15. Dữ liệu json nhận được của thông báo sắp trễ hẹn

**Thông báo có lịch hẹn sắp diễn ra:**

- Thông báo này sẽ gửi thông báo tới các thành viên trong lịch hẹn trước lịch hẹn 1 tiếng.

- API sử dụng : ProcessMsg, ProcessMain

⇒ Thông báo này sẽ được kế thừa từ ProcessMsg và được gọi ở ProcessMain.

```
// Thông báo cuộc họp sắp đến hạn
public static void do_CheckMeetingSoon() {
    Thread t = new Thread() {
        public void run() {
            try {
                Date dtNow = new Date();
                Date dtLim = ToolDate.reqDateByAdding(dtNow, 0, 0, 0, 1, 0, 0);

                List<TaNsoGroup> upcomingMeetings = TaNsoGroup.DAO.reqList(
                    Restrictions.eq(TaNsoGroup.ATT_I_TYPE_01, TaNsoGroup.TYP_01_MEETING),
                    Restrictions.between(TaNsoGroup.ATT_D_DATE_03, dtNow, dtLim)
                );
                for (TaNsoGroup meeting : upcomingMeetings) {
                    reqSaveNotificationMeeting(meeting);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    };
    ThreadManager.doExecuteInfini(t, DefTime.TIME_01_00_00_000, DefTime.TIME_01_00_00_000);
}
```

```
private static void reqSaveNotificationMeeting(TaNsoGroup meeting) throws Exception {
    // Lấy danh sách thành viên của nhóm cuộc họp
    List<TaNsoGroupMember> members = TaNsoGroupMember.DAO.reqList(
        Restrictions.eq(TaNsoGroupMember.COL_I_NSO_GROUP, meeting.reqId())
    );

    JSONArray arr = new JSONArray();
    for (TaNsoGroupMember member : members) {
        JSONObject notify = new JSONObject();
        notify.put("tvpTab", TAB_MEETING);
        notify.put("tvp", TYP_SOON);
        notify.put("title", meeting.req(TaNsoGroup.ATT_T_NAME));
        notify.put("uID", member.req(TaNsoGroupMember.ATT_I_AUT_USER));
        notify.put("parTvp", DefDBExt.ID_TA_NSO_GROUP);
        notify.put("parID", meeting.reqId());
        arr.add(notify);
    }

    reqNewNoti(meeting.reqId(), arr);
}
```

```
//---create noti for task expired/expire soon and meetings
ProcessMsg.do_CheckTaskExpired();
ProcessMsg.do_CheckTaskSoon();
ProcessMsg.do_CheckMeetingSoon();
ProcessMsg.do_CheckPeriodicReminders();|
```



STT	Tên	Dữ liệu được thêm vào		Dữ liệu từ json
1	typ	TYP_SOON	Loại thông báo: gần tới hạn	{ "parTyp": 5000, "uID": 1, "parID": 121, "typTab": "meeting", "typ": 5, "title": "Họp đầu tuần" }
2	typTab	TAB_MEETING	Thông báo về lịch hẹn	
3	title	TaNsoGroup.ATT_T_NAME	Tiêu đề lịch hẹn	
4	uID	TaNsoGroupMember.ATT_I_AUT_USER	ID của người dùng liên quan	
5	parID	meeting.reqId()	ID của đối tượng Nso_Group	
6	parTyp	DefDBExt.ID_TA_NSO_GROUP	ID của đối tượng nhóm (TaNsoGroup). Giá trị này giúp liên kết thông báo với đối tượng chính mà thông báo liên quan đến.	

Hình 5.16. Dữ liệu json nhận được của thông báo lịch hẹn sắp diễn ra.

### c. Lưu thông báo.

```
public static void reqNewNoti(int prjId, JSONArray arrNotify) throws Exception {
    List<TaMsgMessage> lst = new ArrayList<>();

    for (int i = 0; i < arrNotify.size(); i++) {
        JSONObject notifyContent = (JSONObject) arrNotify.get(i);
        TaMsgMessage notif = new TaMsgMessage();
        notif.reqSet(TaMsgMessage.ATT_I_TYPE_01, TaMsgMessage.TYPE_01_NOTIFICATION);
        notif.reqSet(TaMsgMessage.ATT_I_STATUS, TaMsgMessage.STAT_NOTI_NEW);
        notif.reqSet(TaMsgMessage.ATT_D_DATE_01, new Date());
        notif.reqSet(TaMsgMessage.ATT_I_AUT_USER, (Integer)notifyContent.get("uID"));
        notif.reqSet(TaMsgMessage.ATT_T_INFO_01, notifyContent.toString());
        notif.reqSet(TaMsgMessage.ATT_I_ENTITY_TYPE, DefDBExt.ID_TA_PRJ_PROJECT);
        notif.reqSet(TaMsgMessage.ATT_I_ENTITY_ID, prjId);
        lst.add(notif);
    }

    TaMsgMessage.DAO.doPersist(lst);
}
```

### 5.2.3 Kết nối với CSDL.

```
<Resource
name="jdbc/hnv_main"
auth="Container"
type="javax.sql.DataSource"
username="root"
password="260603"
driverClassName="com.mysql.cj.jdbc.Driver"
url="jdbc:mysql://localhost:3306/zino_new"
initialSize="1"
maxWaitMillis="10000"
maxTotal="10"
maxIdle="2"
minIdle="0"
validationQuery="select 1"
poolPreparedStatements="true"
minEvictableIdleTimeMillis="10000"
timeBetweenEvictionRunsMillis="10000"/>
```

Hình 5.17. Code kết nối BE với CSDL

## 5.2.4 Khởi chạy localhoots

```
<?xml version="1.0" encoding="UTF-8"?>
<Server port="8085" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.startup.VersionLoggerListener"/>
  <Listener className="org.apache.catalina.core.AprLifecycleListener"/>
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener"/>
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/>
  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener"/>

  <GlobalNamingResources>
    <Resource auth="Container" description="User database that can be updated and saved" Factory="org.apache.catalina.users.MemoryUserDatabaseFactory" name="UserDatabase" path="/GlobalNamingResources">
    </GlobalNamingResources>

    <Service name="Catalina">
      <Connector compressableMimeType="text/html,text/xml,text/plain,text/css,text/javascript,application/javascript,application/json,https" compression="on" compressionMinSize=
      <Connector port="8080" protocol="AJP/1.3" redirectPort="8443" secretRequired="false"/>

      <Engine defaultHost="localhost" name="Catalina">

        <Realm className="org.apache.catalina.realm.LockOutRealm">
          <Realm className="org.apache.catalina.realm.UserDatabaseRealm" resourceRef="UserDatabase"/>
        </Realm>

        <Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true">
          <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs" pattern="%h %l %u %t &quot;%r&quot;%s %b" prefix="localhost_access_log" suffix=".txt"/>

          <Context docBase="D:/tmp/files" path="/files"/>

          <!-- <Context docBase="IMV_Howline_bo" path="/bo" reloadable="true" source="org.eclipse.jst.jee.server:IMV_Howline_bo"/> -->

          <Context docBase="Zino.be" path="/bo" reloadable="true" source="org.eclipse.jst.jee.server:Zino.be"/>
          <Context docBase="Zino.ui_Bootstrap" path="/" reloadable="true" source="org.eclipse.jst.jee.server:Zino.ui_Bootstrap"/>
        </Host>
      </Engine>
    </Service>
  </Server>
```

Hình 5.18. Code khởi chạy localhoots.



## KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### Đạt được:

- **Kiến thức:**

- + Nắm vững kiến thức về framework HNV-NewLine và SpringBoot, hiểu rõ cơ chế hoạt động và cách áp dụng vào dự án thực tế.
- + Tìm hiểu sâu về cấu trúc và nguyên lý hoạt động của hệ thống quản lý dự án Zino.
- + Nghiên cứu và áp dụng các phương pháp thiết kế và phát triển phần mềm hiện đại.

- **Kỹ năng:**

- + Phát triển: Thành thạo trong việc xây dựng các module thông báo, tích hợp với các thành phần khác của hệ thống.
- + Lập trình: Nâng cao khả năng viết code sạch, hiệu quả, tuân thủ các quy tắc coding convention.
- + Làm việc nhóm: Tham gia tích cực vào các cuộc họp, thảo luận, đóng góp ý kiến để hoàn thành dự án.
- + Quản lý thời gian: Lên kế hoạch và thực hiện công việc một cách hiệu quả, đảm bảo đúng tiến độ.

- **Khác:**

- + Tích lũy kinh nghiệm thực tế trong môi trường làm việc chuyên nghiệp.
- + Rèn luyện tư duy logic, khả năng giải quyết vấn đề và làm việc độc lập.

### Hạn chế:

- **Kiến thức:** Chưa hiểu sâu về các công nghệ mới, đặc biệt là các thư viện và framework liên quan đến phát triển ứng dụng web.

- **Kỹ năng:** Cần cải thiện khả năng tối ưu hóa hiệu suất của ứng dụng, viết các đoạn code phức tạp hơn.

- **Khác:** Chưa có nhiều kinh nghiệm trong việc triển khai ứng dụng lên môi trường sản xuất.

## Hướng phát triển:

- **Ngắn hạn:**

- + Tìm hiểu và làm quen với các công nghệ mới như React, Angular,....
- + Tham gia các buổi meetup, hội thảo về lập trình để kết nối với các developer khác, học hỏi kinh nghiệm.
- + Nghiên cứu các hệ quản trị cơ sở dữ liệu khác nhau (SQL, NoSQL) để lựa chọn công cụ phù hợp với từng dự án.
- + Học cách thiết kế cơ sở dữ liệu hiệu quả, tối ưu hóa truy vấn.
- + Tìm tòi các giải pháp tối ưu hóa hiệu năng, bảo mật cho các ứng dụng.

- **Dài hạn:**

- + Trở thành một Fullstack Developer chuyên nghiệp, có khả năng xây dựng các ứng dụng web phức tạp.
- + Áp dụng những kiến thức đã học vào các dự án thực tế để củng cố và nâng cao kỹ năng.
- + Đóng góp vào việc phát triển các sản phẩm phần mềm có giá trị cho cộng đồng.

## TÀI LIỆU THAM KHẢO

- [1] "Tra tên công ty," [Online]. Available:  
<https://www.tratencongtty.com/company/511c8e7c-cong-ty-co-phan-inotev/>.
- [2] "INOTEV," công ty cổ phần INOTEV, [Trực tuyến]. Available:  
<https://www.inotev.fr/>.
- [3] "blog," glints, [Online]. Available: <https://glints.com/vn/blog/full-stack-la-gi/>.
- [4] "blog," itviec, [Online]. Available: <https://itviec.com/blog/lap-trinh-full-stack-la-gi/>.
- [5] "blog," fullstack.edu.vn, [Online]. Available: <https://fullstack.edu.vn/blog/fullstack-la-gi-can-ky-nang-gi-de-tro-thanh-fullstack-developer>.
- [6] "blog," topdev, [Online]. Available: <https://topdev.vn/blog/gioi-thieu-ve-spring-boot-spring-boot-la-gi/>.
- [7] "wiki," wikipedia, [Online]. Available:  
[https://en.wikipedia.org/wiki/Multitier\\_architecture](https://en.wikipedia.org/wiki/Multitier_architecture).
- [8] "blog," 200lab, [Online]. Available: <https://200lab.io/blog/gioi-thieu-microservice/>.
- [9] "viblo," viblo, [Online]. Available: <https://viblo.asia/p/hexagonal-architecture-la-gi-va-ung-dung-cua-no-4dbZNR88ZYM>.
- [10] "blog," vn.got-it.ai, [Online]. Available: <https://vn.got-it.ai/blog/eclipse-la-gi-huong-dan-cai-dat-eclipse-chi-tiet-nhat>.
- [11] "blog," codegym, [Online]. Available: <https://codegym.vn/blog/hoc-ngon-ngu-java-co-ban-tu-a-z/>.
- [12] "blog," topdev, [Online]. Available: <https://topdev.vn/blog/gioi-thieu-ve-mysql/>.

## CHECK LIST CỦA BÁO CÁO

STT	Nội dung công việc	Có	Không	Ghi chú
1	Báo cáo được trình bày (định dạng) đúng với yêu cầu.	x		
2	Báo cáo có số lượng trang đáp ứng đúng yêu cầu (30-60 trang)	x		
3	Báo cáo trình bày được phần mở đầu bao gồm: Mục tiêu, Phạm vi và đối tượng, kết cấu ...	x		
4	Báo cáo trình bày về công ty, vị trí việc làm (công việc đó làm gì, kiến thức và kỹ năng cần thiết là gì, con đường phát triển sự nghiệp (career path)), cơ sở lý thuyết phù hợp với nội dung của đề tài (Tối đa 10-12 trang)	x		
5	Báo cáo có sản phẩm cụ thể phù hợp với mục tiêu đặt ra của đề tài	x		
6	Báo cáo có phần kết luận và hướng phát triển của đề tài	x		

## PHỤ LỤC

Link video demo: [Demo](#)

Link code: [Code](#)

