

**BỘ GIÁO DỤC VÀ ĐÀO TẠO  
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG  
KHOA CÔNG NGHỆ 2**



**MÔN HỌC: KIẾN TRÚC VÀ THIẾT KẾ PHẦN MỀM**

**Đề tài:**

***XÂY DỰNG HỆ THỐNG HỖ TRỢ GIAO VIỆC VÀ  
THEO DÕI TIẾN ĐỘ CÔNG VIỆC NHÓM***

*Lớp: D21CQCNPM01-N.*

*Nhóm: 29*

*Giảng viên hướng dẫn: Thầy Nguyễn Văn Hữu Hoàng*

*Sinh viên thực hiện: Lương Thành Lợi – N21DCCN050*

*Nguyễn Thành Trung – N21DCCN087*

*Nguyễn Anh Tuấn – N21DCCN090*

*Thành phố Hồ Chí Minh, tháng 6 năm 2025*

## MỤC LỤC

LỜI CẢM ƠN .....	1
BẢNG PHÂN CÔNG .....	2
I. MỤC ĐÍCH YÊU CẦU .....	3
1. Mục đích (Nhu cầu sử dụng ứng dụng) .....	3
2. Đề tài “Xây dựng hệ thống hỗ trợ giao việc và theo dõi tiến độ công việc nhóm” được triển khai nhằm: .....	3
3. Yêu cầu.....	4
4. Sơ đồ use case .....	5
5. Biểu đồ tuần tự - Sequence Diagram .....	6
6. Vẽ biểu đồ hoạt động .....	15
7. Sơ đồ lớp .....	29
II. THỰC HIỆN .....	30
1. Công nghệ sử dụng: .....	30
2. Mô hình kiến trúc:.....	30
3. Các mẫu Design Pattern sử dụng .....	31
Singleton Design Pattern .....	31
Template Method .....	31
Adapter Pattern.....	31
III. THIẾT KẾ GIAO DIỆN .....	33
VI. KẾT LUẬN.....	38

## **LỜI CẢM ƠN**

Lời đầu tiên, nhóm em xin gửi lời cảm ơn chân thành đến Thầy Nguyễn Văn Hữu Hoàng. Trong quá trình học tập và nghiên cứu bộ môn Chuyên đề Công Nghệ Phần mềm, chúng em đã nhận được sự hướng dẫn, giảng dạy tận tâm và nhiệt huyết từ thầy. Thầy đã giúp chúng em hiểu rõ hơn về những kiến thức quan trọng và thực tế trong lĩnh vực quản lý dự án, từ đó tích lũy thêm nhiều kinh nghiệm bổ ích và kỹ năng cần thiết.

Từ những kiến thức mà thầy truyền đạt, nhóm em đã áp dụng vào việc hoàn thành bài tiểu luận cuối kỳ của môn học. Tuy nhiên, do vẫn còn những hạn chế về kinh nghiệm và hiểu biết trong lĩnh vực này, bài tiểu luận của chúng em khó tránh khỏi những thiếu sót. Rất mong thầy xem xét, đánh giá và góp ý để bài tiểu luận của chúng em được hoàn thiện hơn.

Nhóm em kính chúc thầy luôn dồi dào sức khỏe, thành công trong sự nghiệp giảng dạy, và tiếp tục dìu dắt các thế hệ sinh viên đến với những thành tựu mới trên con đường học tập và phát triển bản thân. Nhóm em xin chân thành cảm ơn!

## **BẢNG PHÂN CÔNG**

<b>Tên thành viên</b>	<b>Nhiệm vụ</b>
Lương Thành Lợi	<ul style="list-style-type: none"><li>- Các chức năng liên quan đến tìm kiếm, thông báo, thống kê.</li><li>- Trình bày báo cáo.</li></ul>
Nguyễn Thành Trung	<ul style="list-style-type: none"><li>- Các chức năng liên quan đến account, task.</li><li>- Vẽ các biểu đồ</li></ul>
Nguyễn Anh Tuấn	<ul style="list-style-type: none"><li>- Các chức năng liên quan đến project, phase, xác thực người dùng.</li><li>- Phân tích yêu cầu, chức năng.</li></ul>

# I. MỤC ĐÍCH YÊU CẦU

## ***1. Mục đích (Nhu cầu sử dụng ứng dụng)***

Trong bối cảnh hiện nay, việc phân công và theo dõi công việc nhóm thường diễn ra qua đa kênh như email, chat hay giấy tờ, dẫn đến thiếu minh bạch, khó nắm bắt tiến độ và dễ bỏ sót hoặc trùng lặp nhiệm vụ. Người quản lý phải mất nhiều thời gian để tổng hợp thông tin, còn thành viên dễ nhầm lẫn hoặc không biết rõ trách nhiệm. Do đó, cần một công cụ tập trung, giúp mọi người phối hợp rõ ràng, nhanh chóng và đồng bộ từ khâu giao việc đến khi hoàn thành.

Mục đích của đề tài là xây dựng một hệ thống trực tuyến cho phép trưởng nhóm tạo, mô tả công việc, đặt deadline, ưu tiên và giao thẳng cho từng thành viên; đồng thời, theo dõi tiến độ theo thời gian thực. Thành viên có thể cập nhật trạng thái trao đổi và đính kèm tài liệu ngay trên từng đầu việc. Nhờ vậy, thông tin luôn minh bạch, quản lý kịp thời can thiệp khi công việc chậm tiến độ hoặc vướng mắc.

Hệ thống cũng cung cấp không gian bình luận cho mỗi công việc, giúp thành viên và quản lý trao đổi nhanh chóng mà không phải chuyển qua nhiều kênh khác. Tính năng thông báo tức thì đảm bảo mọi người không bỏ lỡ thông tin quan trọng. Tổng hợp tình trạng công việc, phân bổ khối lượng và tính sơ bộ KPI đơn giản, đáp ứng nhu cầu báo cáo lên cấp trên hoặc khách hàng.

Việc triển khai giải pháp này đem lại nhiều lợi ích: tăng trách nhiệm cá nhân, giảm thời gian họp hành, hạn chế sai sót khi phân công và tránh chồng chéo công việc. Mọi thông tin, bình luận và tệp đính kèm đều tập trung tại một nơi, giúp cải thiện chất lượng trao đổi và nâng cao hiệu quả chung của nhóm.

## ***2. Đề tài “Xây dựng hệ thống hỗ trợ giao việc và theo dõi tiến độ công việc nhóm” được triển khai nhằm:***

- Phát triển một hệ thống trực tuyến cho phép người dùng (trưởng nhóm, thành viên) tạo, giao và cập nhật tiến độ công việc một cách nhanh chóng, minh bạch.
- Hỗ trợ quản lý các thông tin liên quan đến dự án, công việc, thành viên, trạng thái công việc, bình luận và file đính kèm.
- Cung cấp giao diện quản trị để trưởng nhóm duyệt/tái phân công công việc, theo dõi tiến độ tổng thể và kiểm soát khối lượng công việc của từng thành viên.
- Ứng dụng kiến trúc phần mềm hiện đại (MVC, Layered Architecture) cùng các mẫu thiết kế (Design Patterns) nhằm đảm bảo tính mở rộng, dễ bảo trì và tối ưu hiệu năng của hệ thống.

- Đảm bảo hệ thống hoạt động ổn định, an toàn về dữ liệu và thân thiện với người sử dụng cuối.

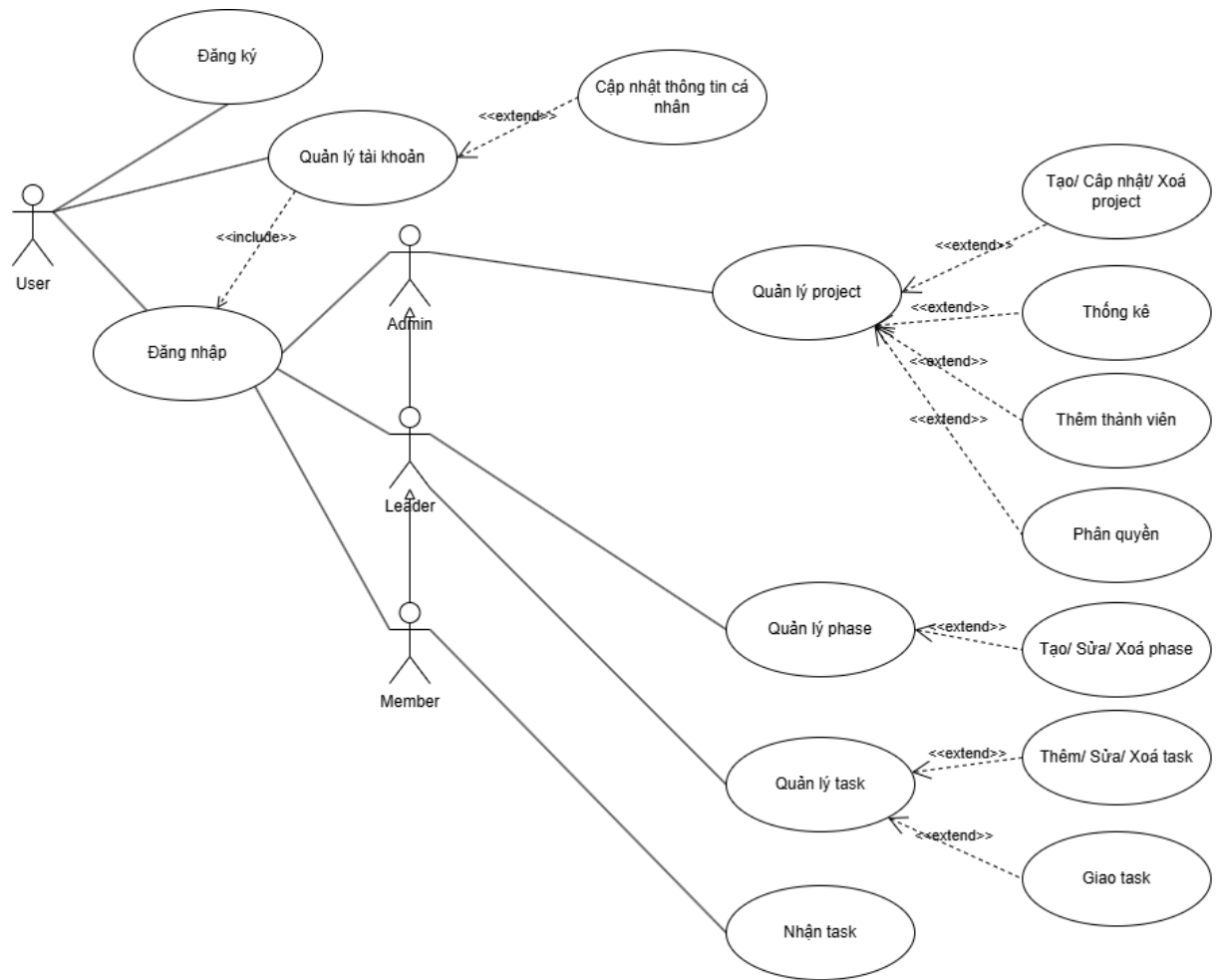
### 3. Yêu cầu

Hệ thống phải cho phép đăng ký/đăng nhập, quản lý dự án và công việc: tạo, sửa, xóa, phân công, cập nhật trạng thái, bình luận, đính kèm file và hỗ trợ tìm kiếm, lọc, sắp xếp task theo dự án, deadline, ưu tiên hoặc người thực hiện, phân quyền (ADMIN, LEADER, MEMBER). Khi có thay đổi trạng thái hoặc task quá hạn, hệ thống cần gửi thông báo tức thì (email/notification) và hiển thị badge trên giao diện. Đồng thời, cần có dashboard tổng quan với biểu đồ tiến độ, bảng phân bổ khối lượng công việc và khả năng xuất báo cáo (Excel/PDF) theo tuần hoặc tháng. Giao diện phải responsive, phản hồi nhanh (<300 ms). Về kỹ thuật, ứng dụng kiến trúc MVC, sử dụng JWT cho xác thực, mã hóa mật khẩu (bcrypt), validate đầu vào, chống CSRF/XSS, Mã nguồn phải áp dụng các Design Patterns để đảm bảo tính mở rộng, dễ bảo trì và khả năng mở rộng tích hợp.

Phân tích và thiết kế hệ thống:

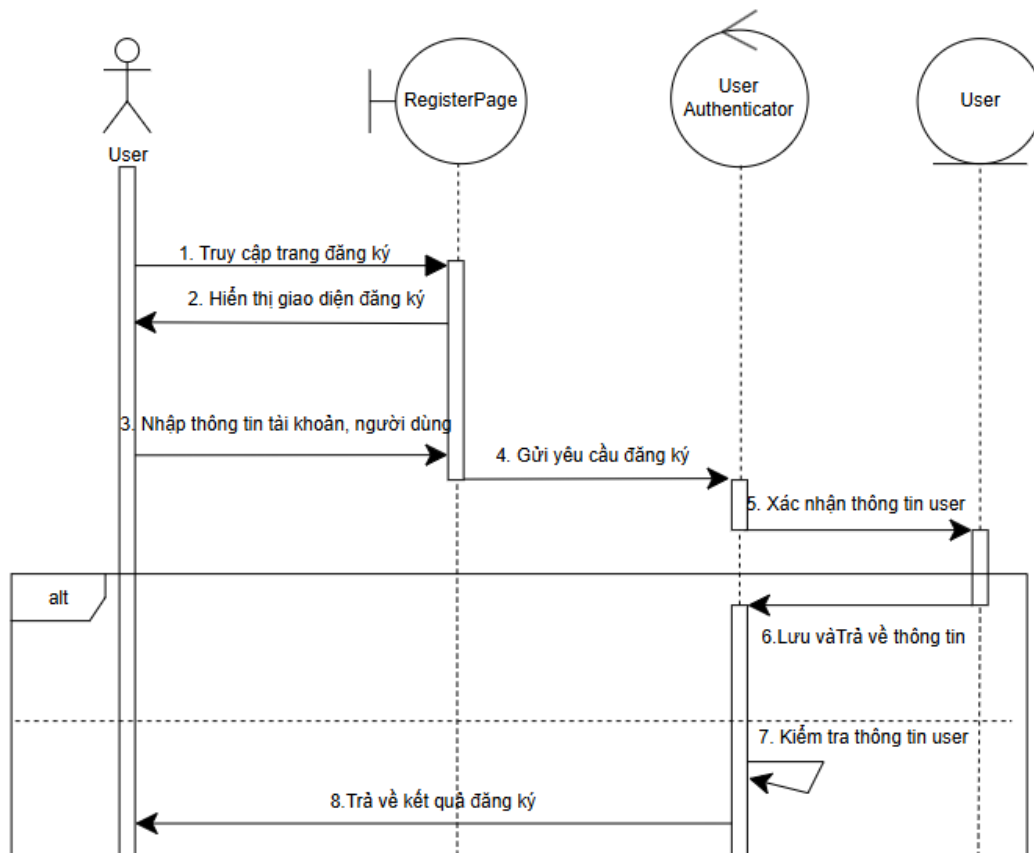
- Vẽ sơ đồ Use Case, Biểu đồ lớp (Class Diagram), Biểu đồ tuần tự (Sequence Diagram), Biểu đồ hoạt động (Activity Diagram) bằng UML.
- Thiết kế giao diện người dùng thân thiện, đảm bảo UX tốt (Responsive UI với PrimeFaces).
- Thiết kế cơ sở dữ liệu My SQL chuẩn hóa theo mô hình thực thể - quan hệ (ERD).

#### 4. Sơ đồ use case



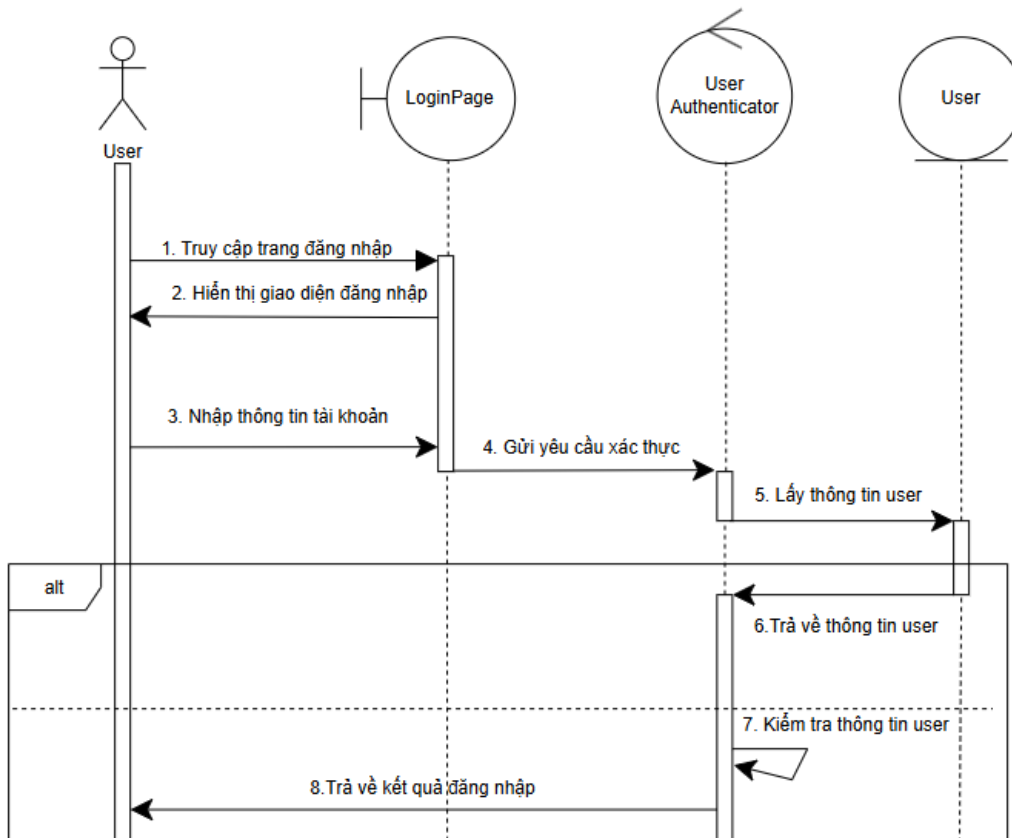
## 5. Biểu đồ tuần tự - Sequence Diagram

### 6.1. Cas sử dụng “Đăng kí”

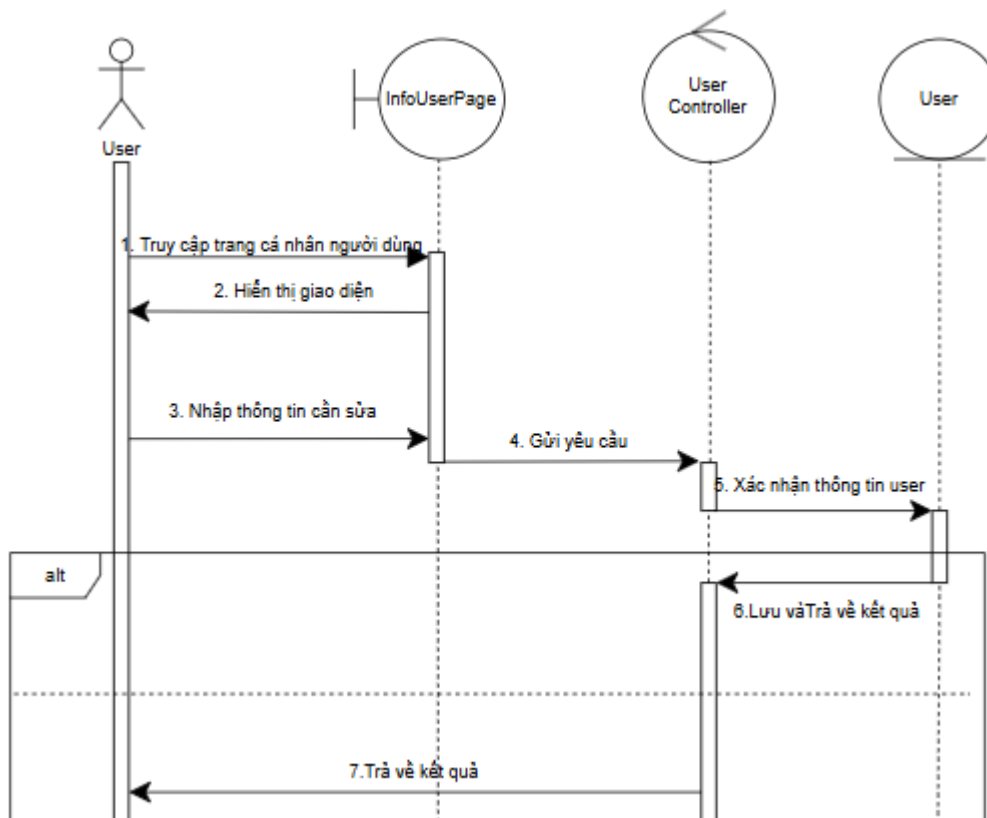




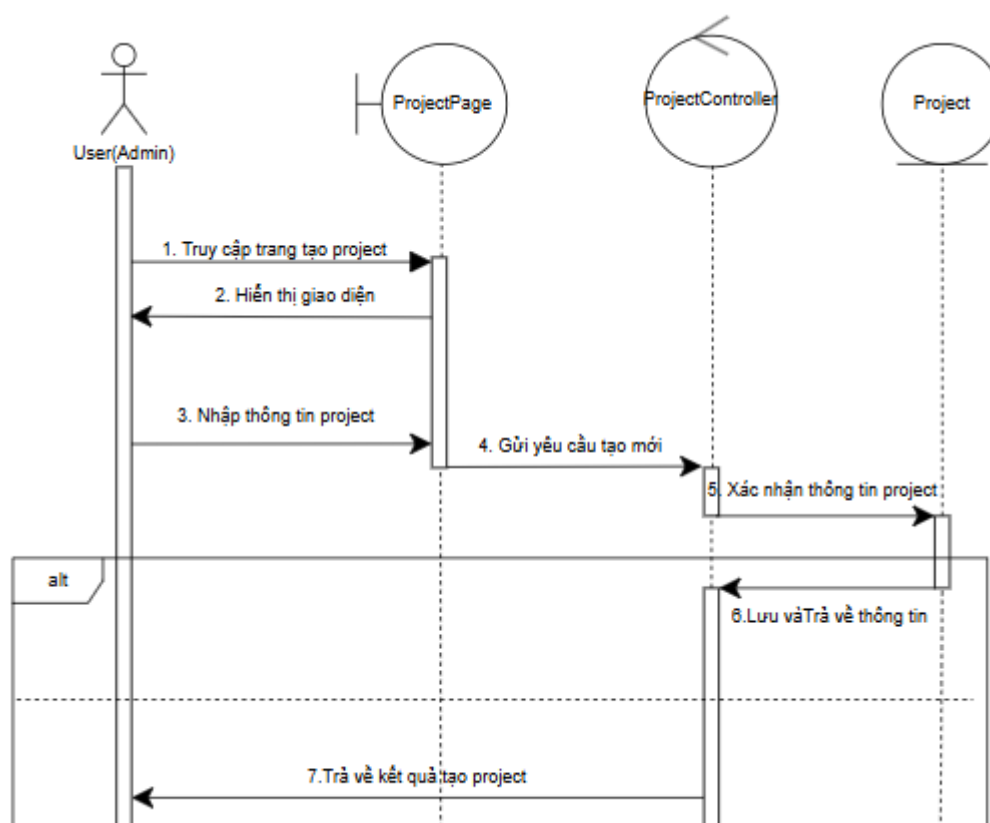
## 6.2. Ca sử dụng “ Đăng nhập ”



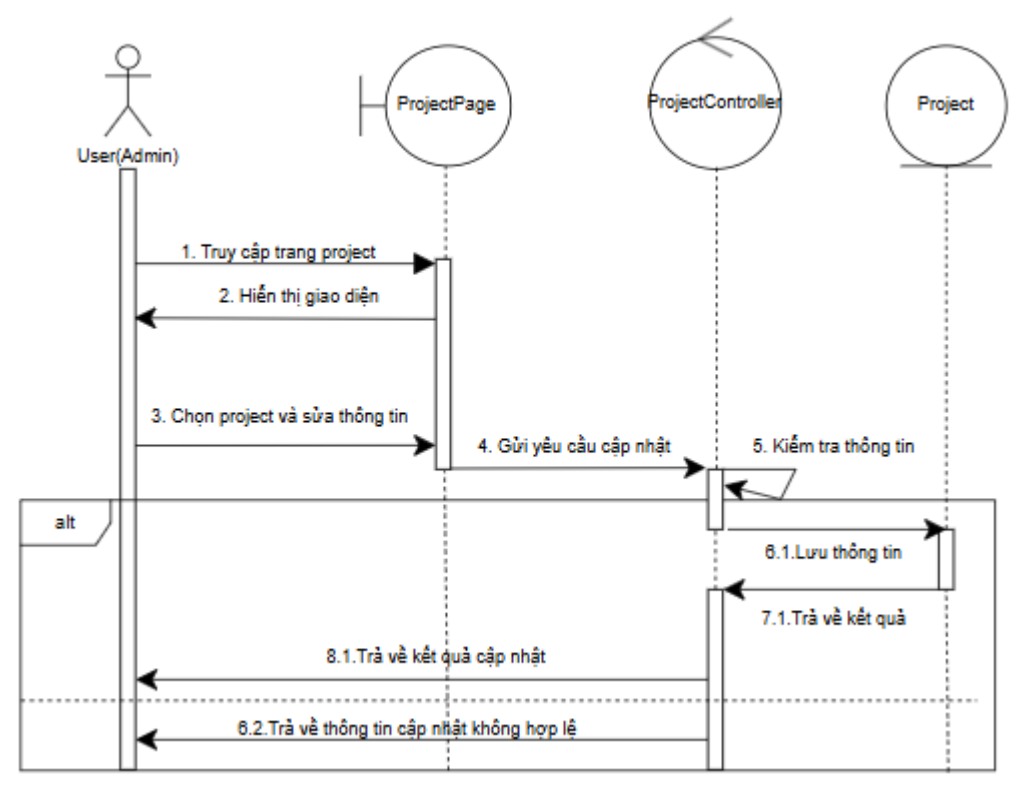
## 6.3. Ca sử dụng “ Cập nhật thông tin cá nhân ”



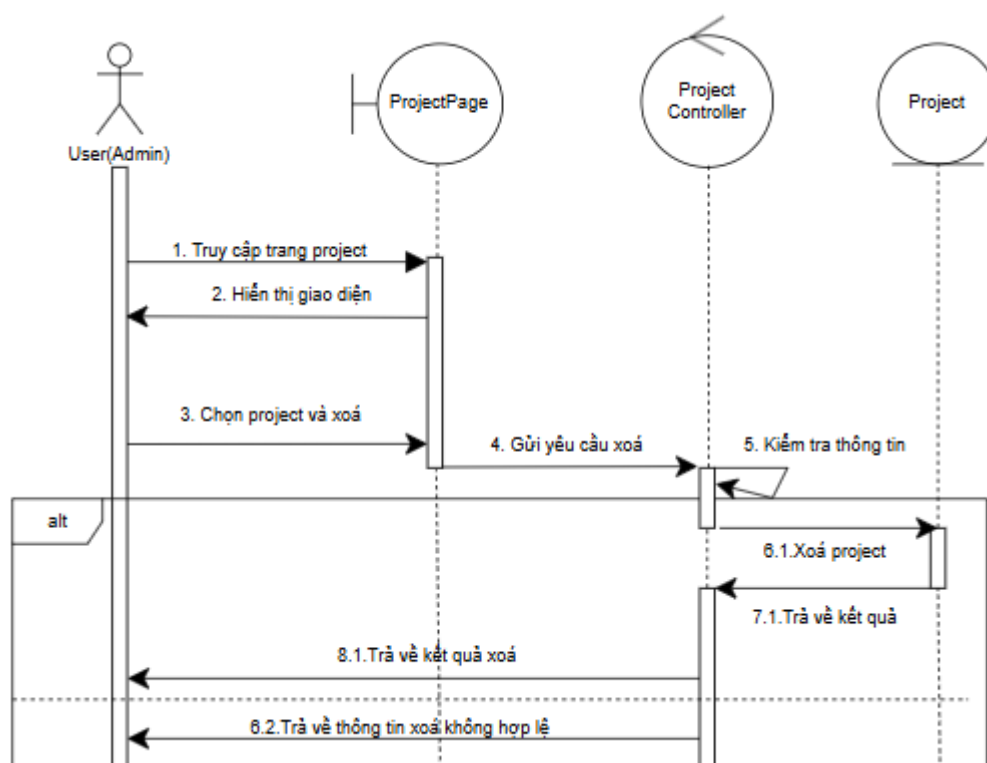
#### 6.4. Ca sử dụng “Tạo project”



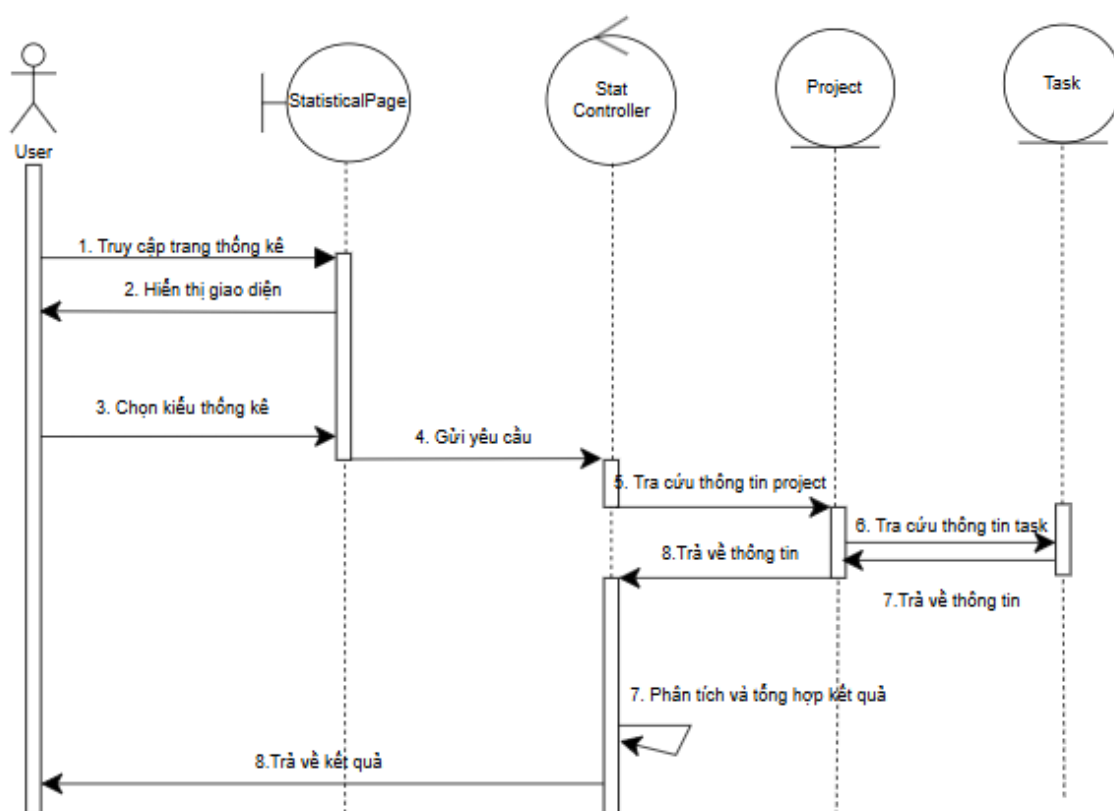
#### 6.5. Ca sử dụng “Cập nhật project”



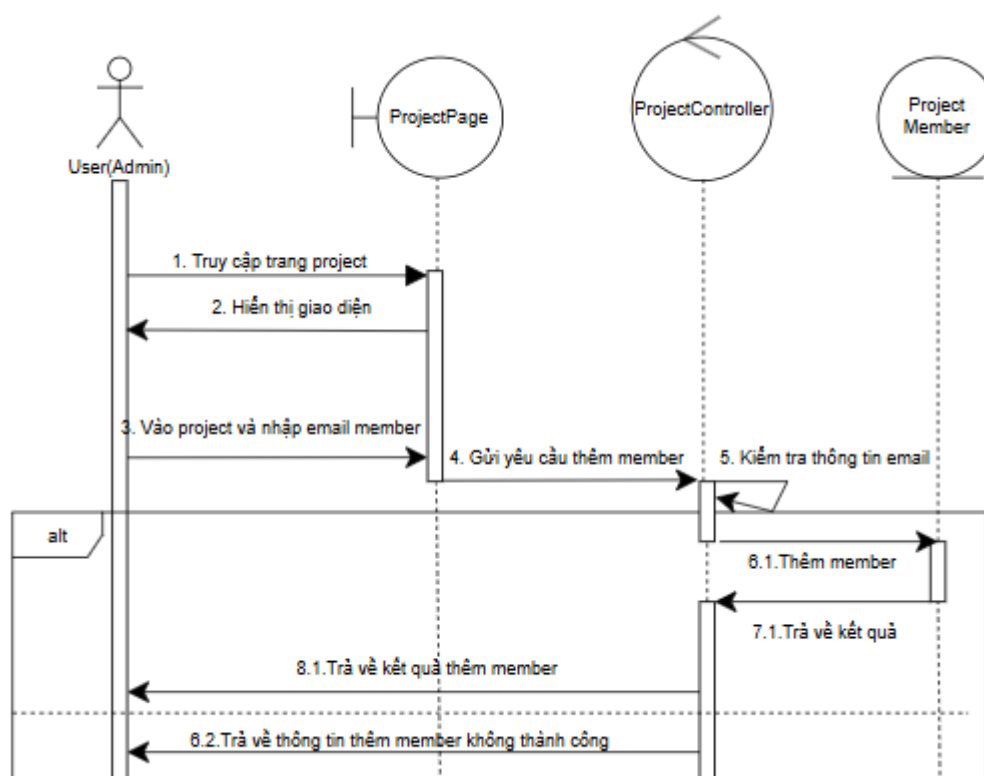
### 6.6. Ca sử dụng “Xoá project”



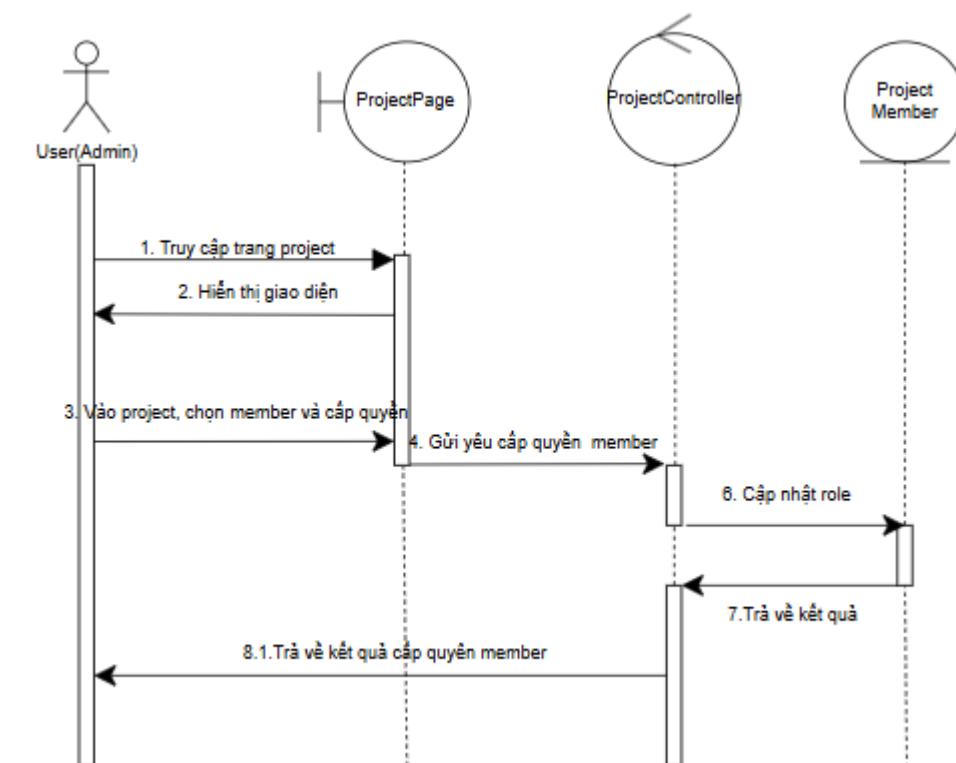
### 6.7. Ca sử dụng “Thống kê”



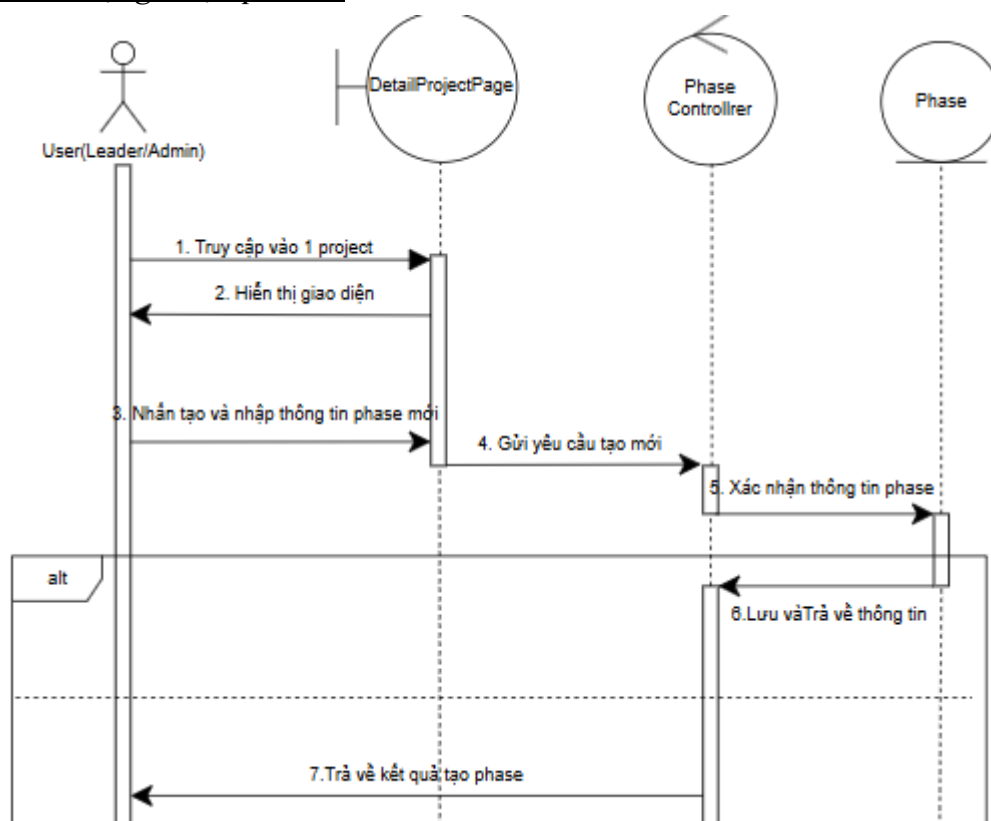
### 6.8. Ca sử dụng “Thêm thành viên”



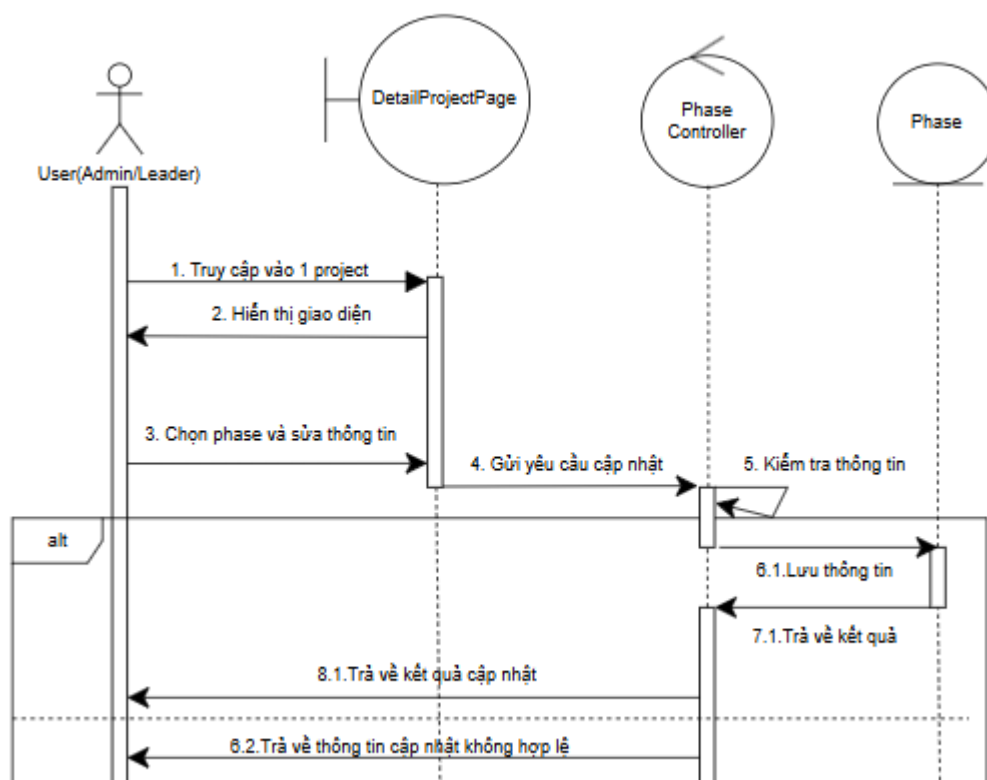
### 6.9. Ca sử dụng “Phân quyền”



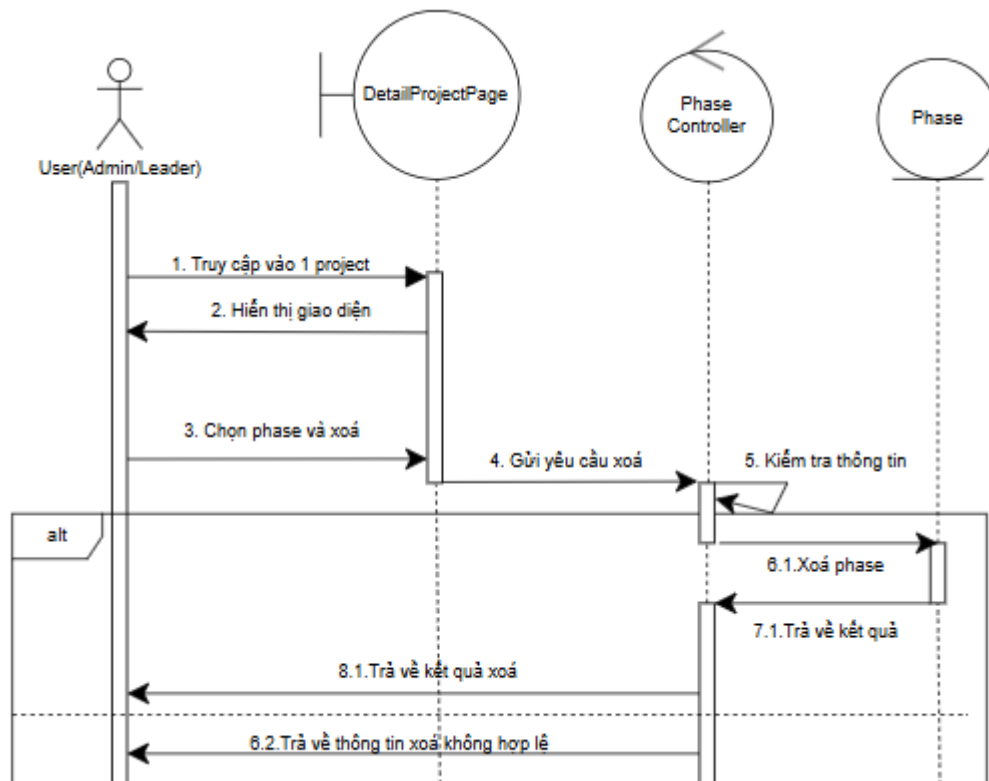
### 6.10. Ca sử dụng “Tạo phase”



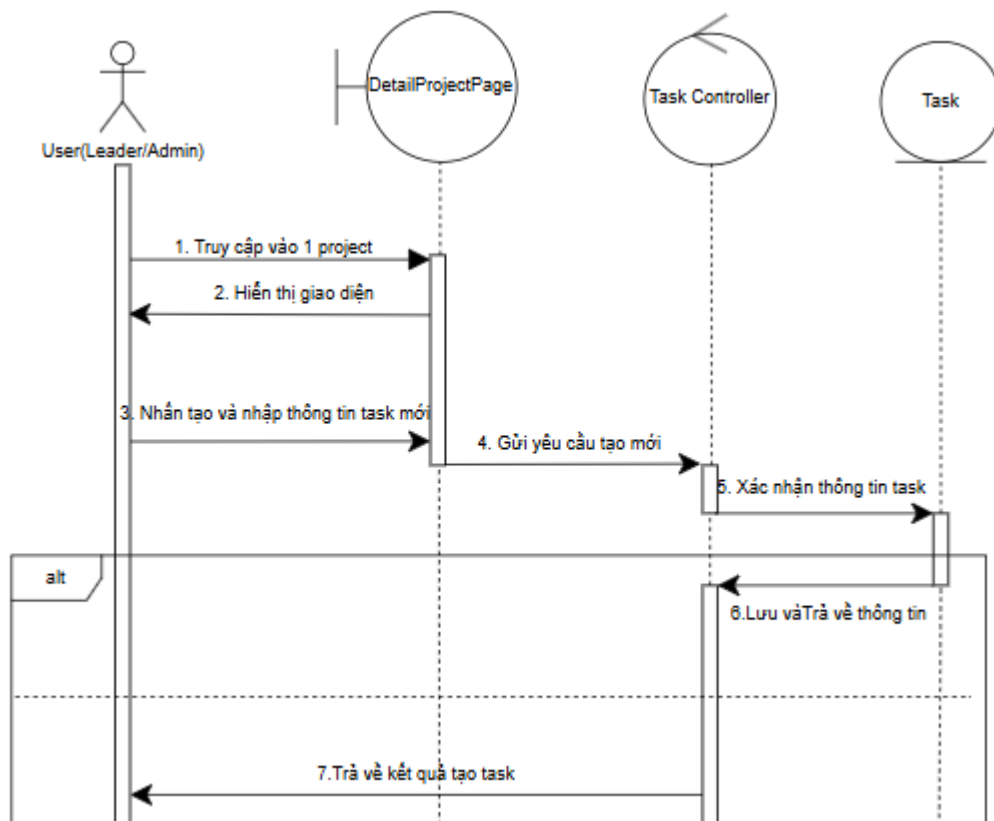
### 6.11. Ca sử dụng “Cập nhật phase”



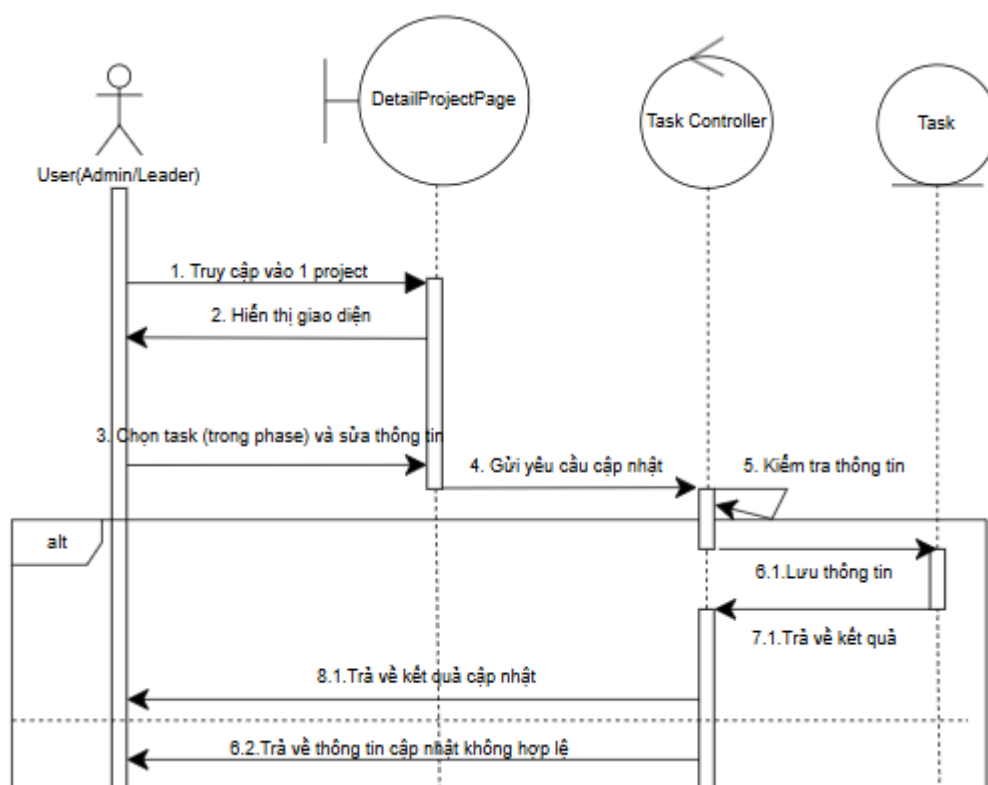
### 6.12. Casử dụng “Xoá phase”



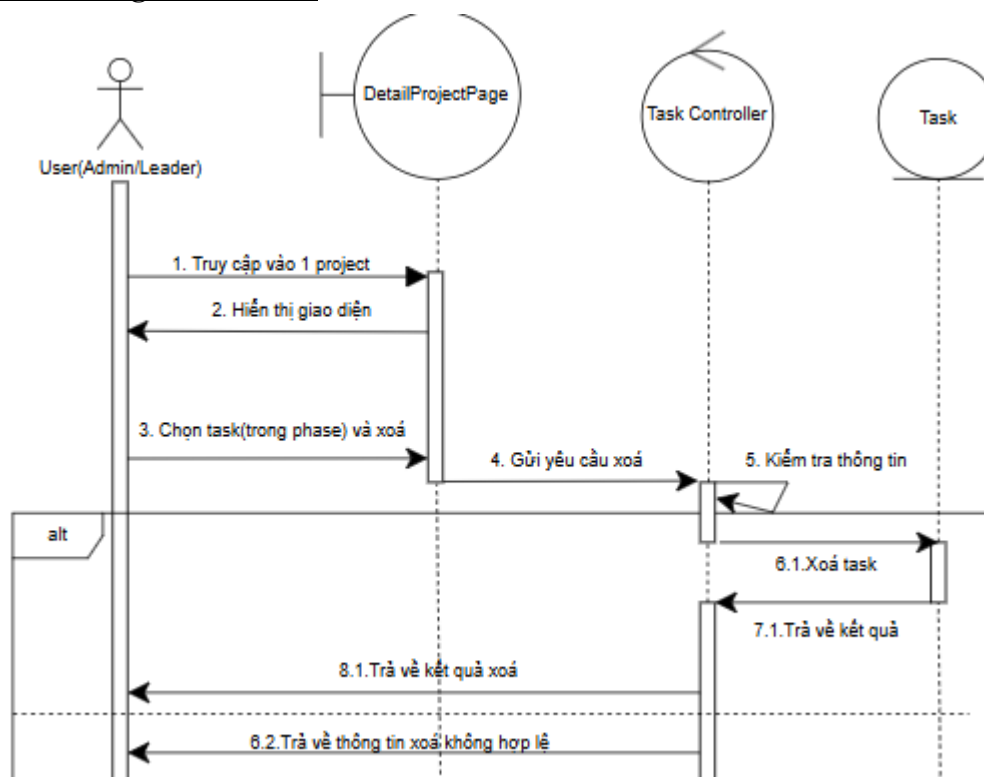
### 6.13. Casử dụng “Tạo task”



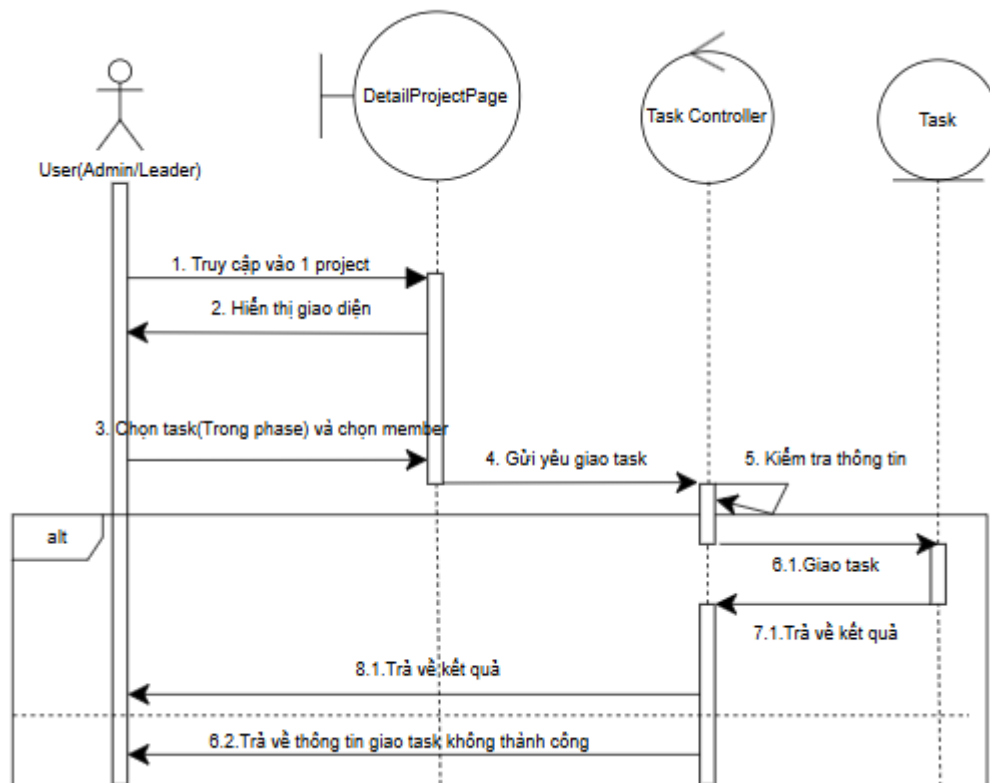
#### 6.14. Casử dụng “Cập nhật task”



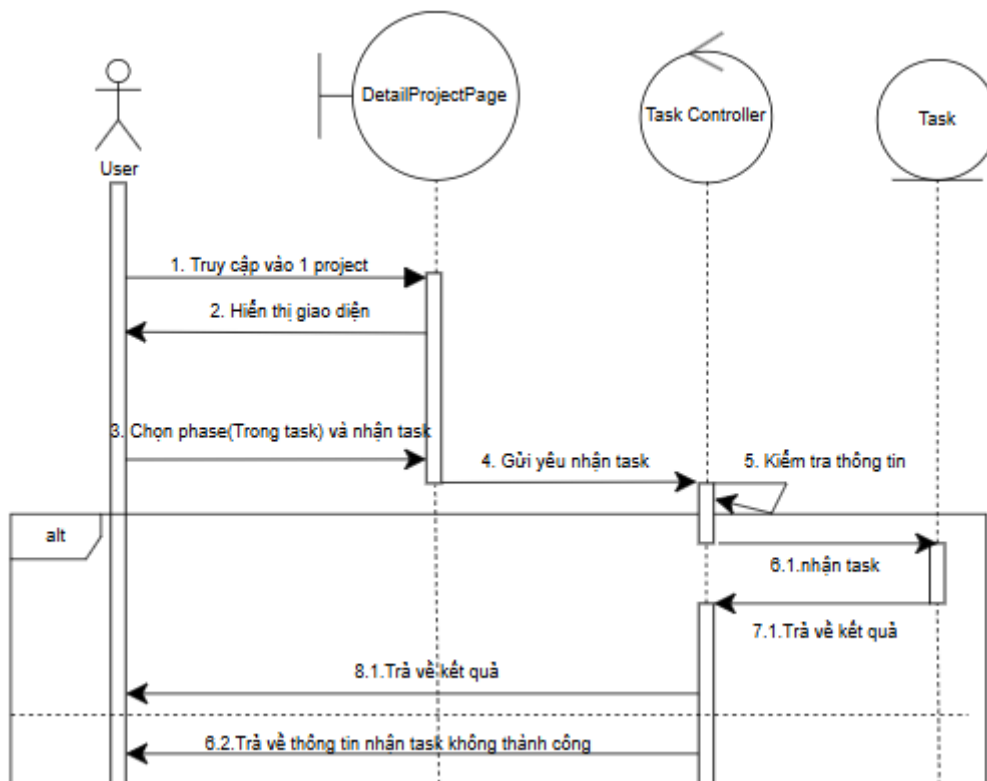
#### 6.13. Casử dụng “Xoá task”



### 6.13. Cả sử dụng “Giao task”



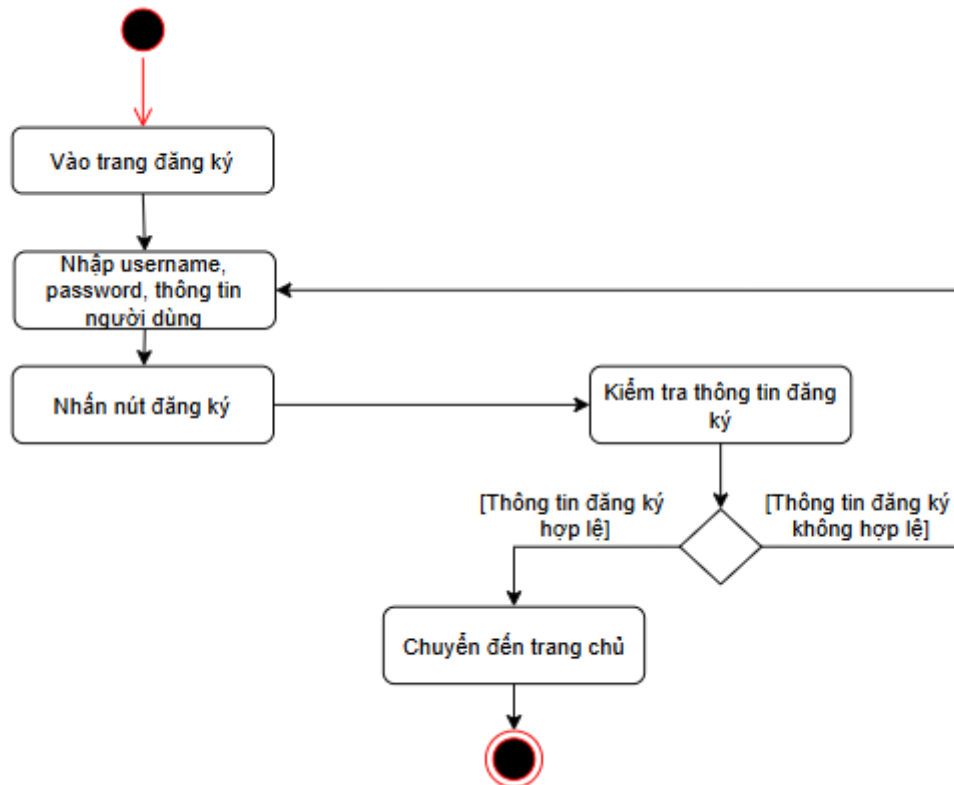
### 6.14. Cả sử dụng “Nhận task”



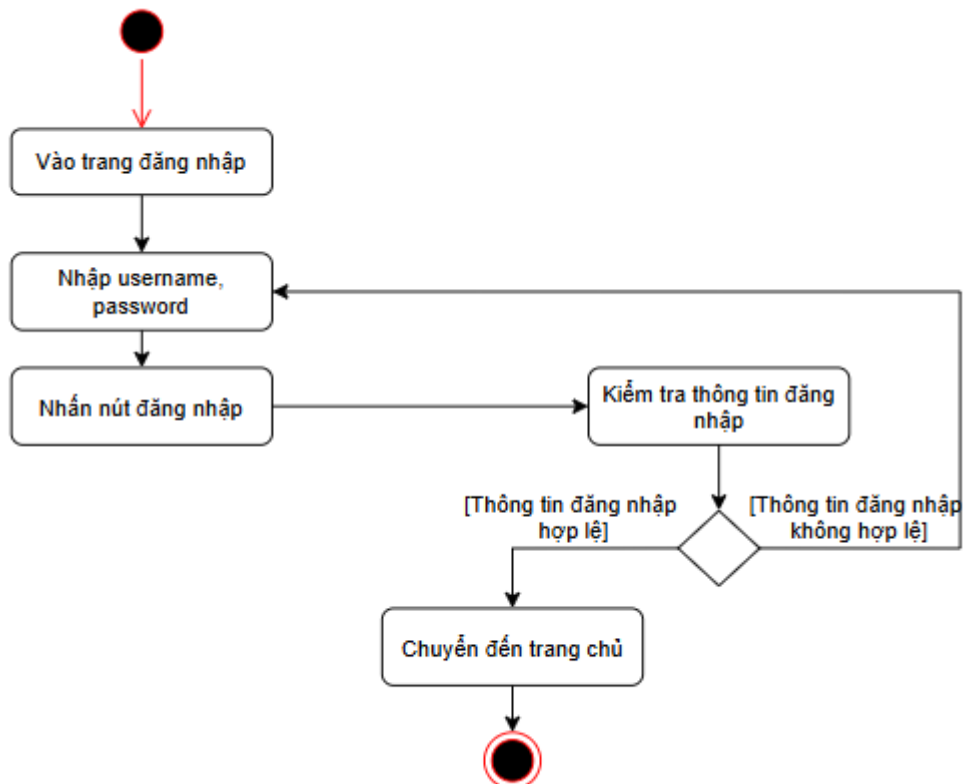


## 6. Vẽ biểu đồ hoạt động

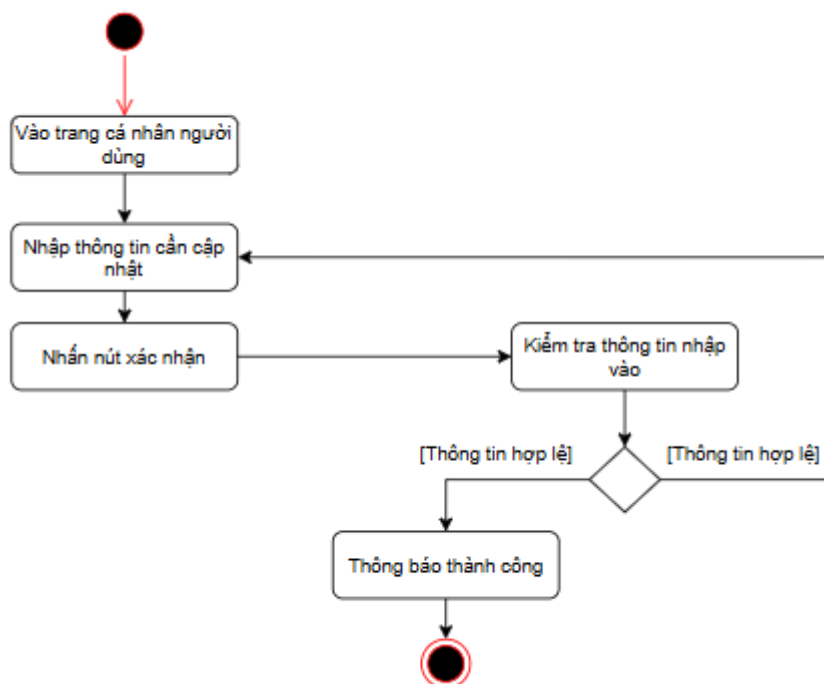
### 7.1. Ca sử dụng “ Đăng kí ”



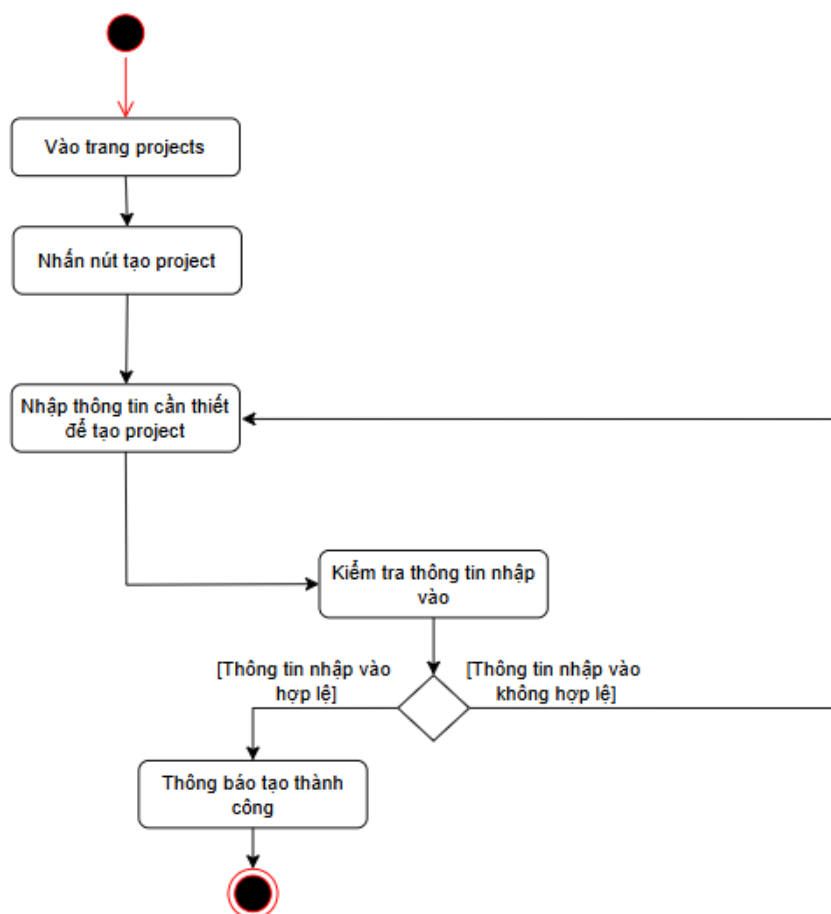
### 7.2. Ca sử dụng “ Đăng nhập ”



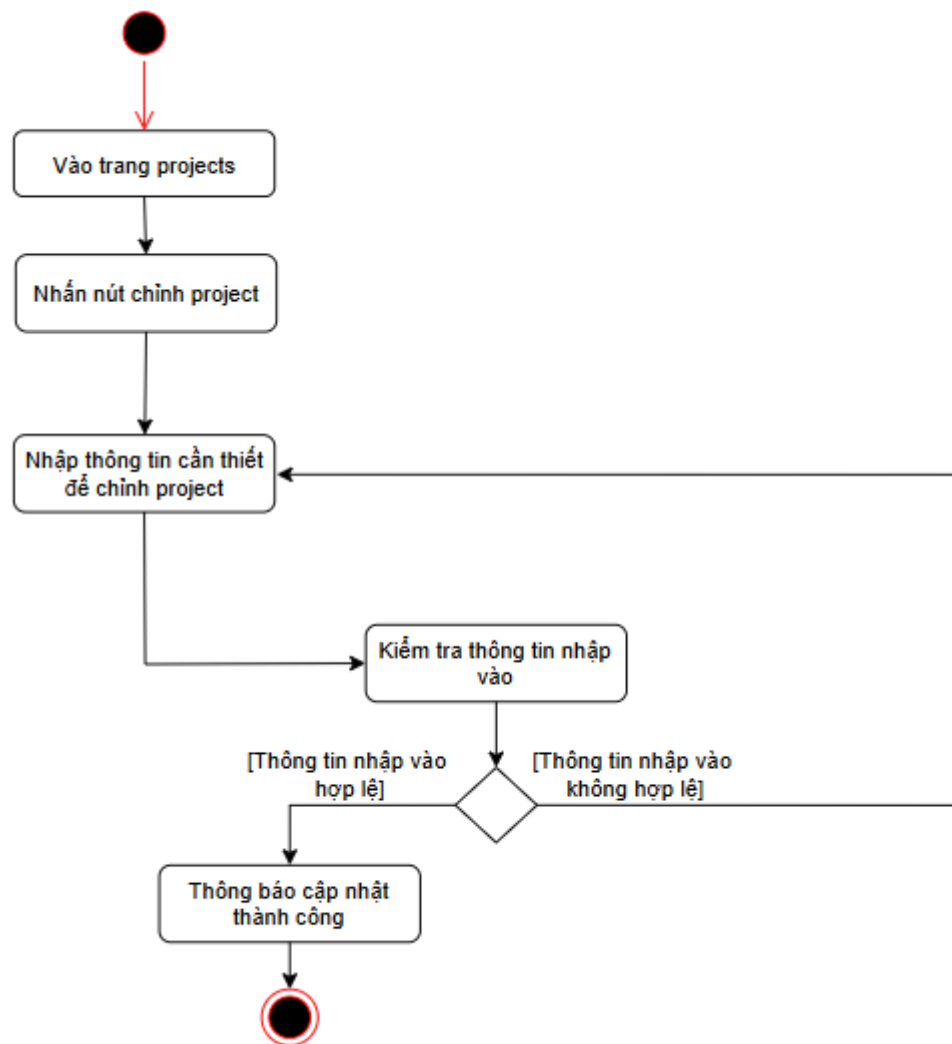
### 7.3. Ca sử dụng “Cập nhật thông tin cá nhân”



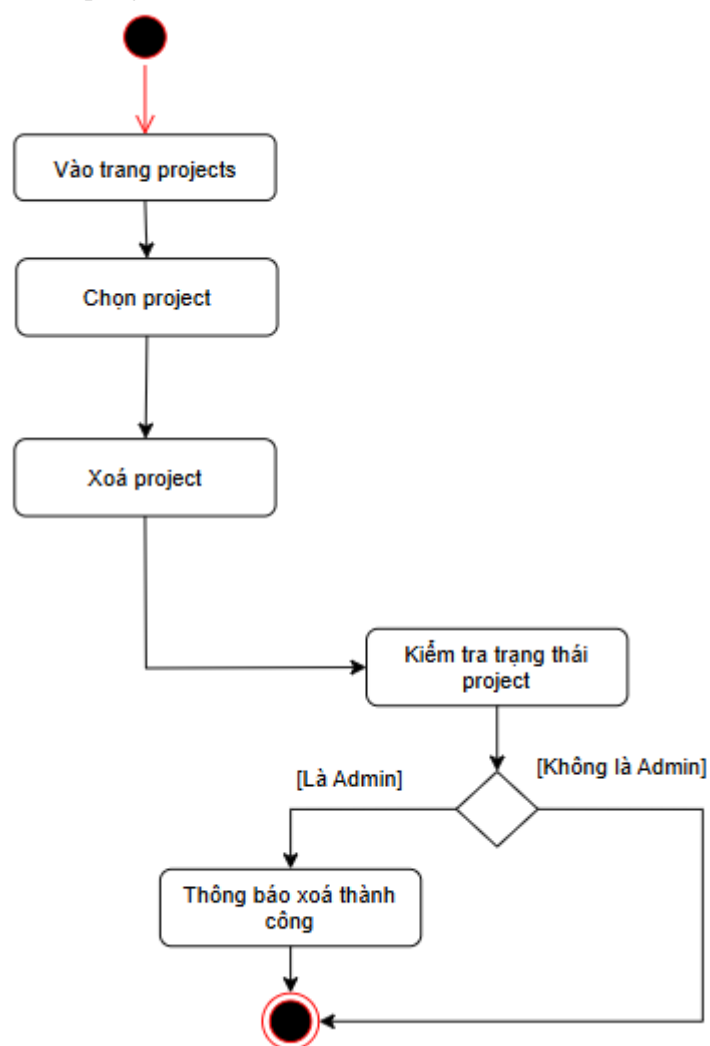
### 7.4. Ca sử dụng “Tạo project”



### 7.5. Ca sử dụng “ Cập nhật project “



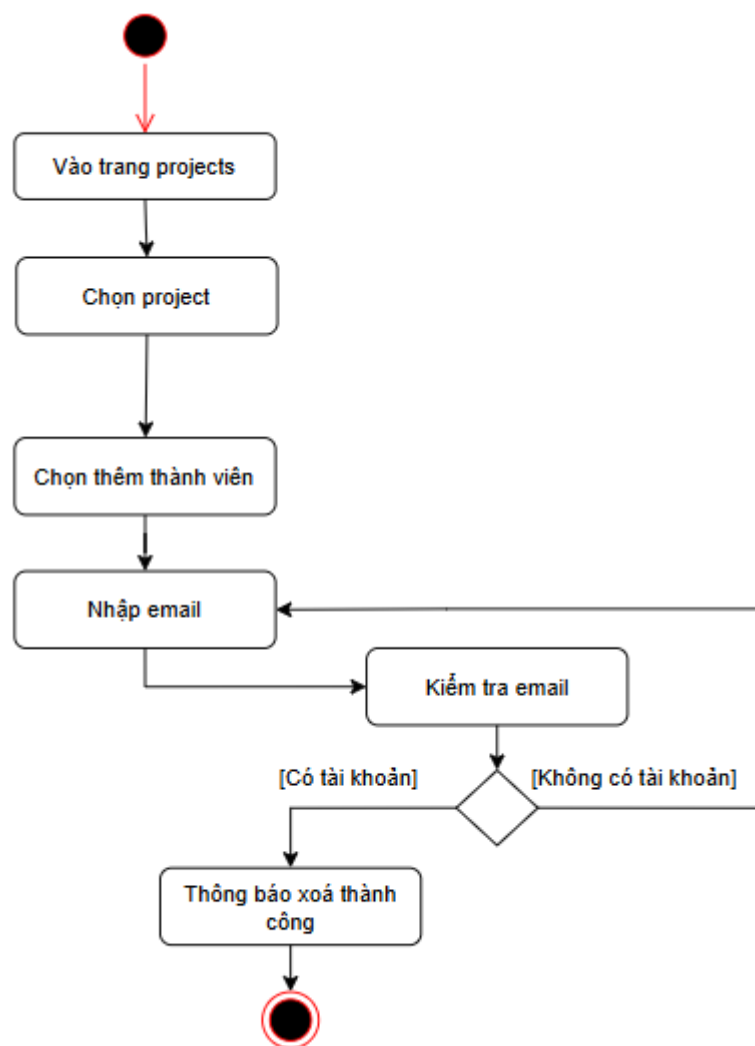
### 7.6. Ca sử dụng “Xoá project”



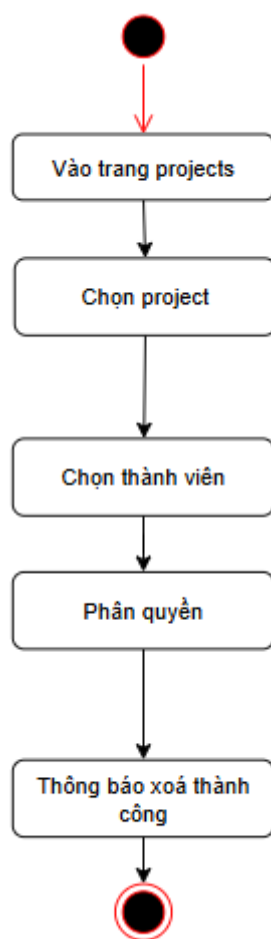
### 7.7. Ca sử dụng “Thống kê”



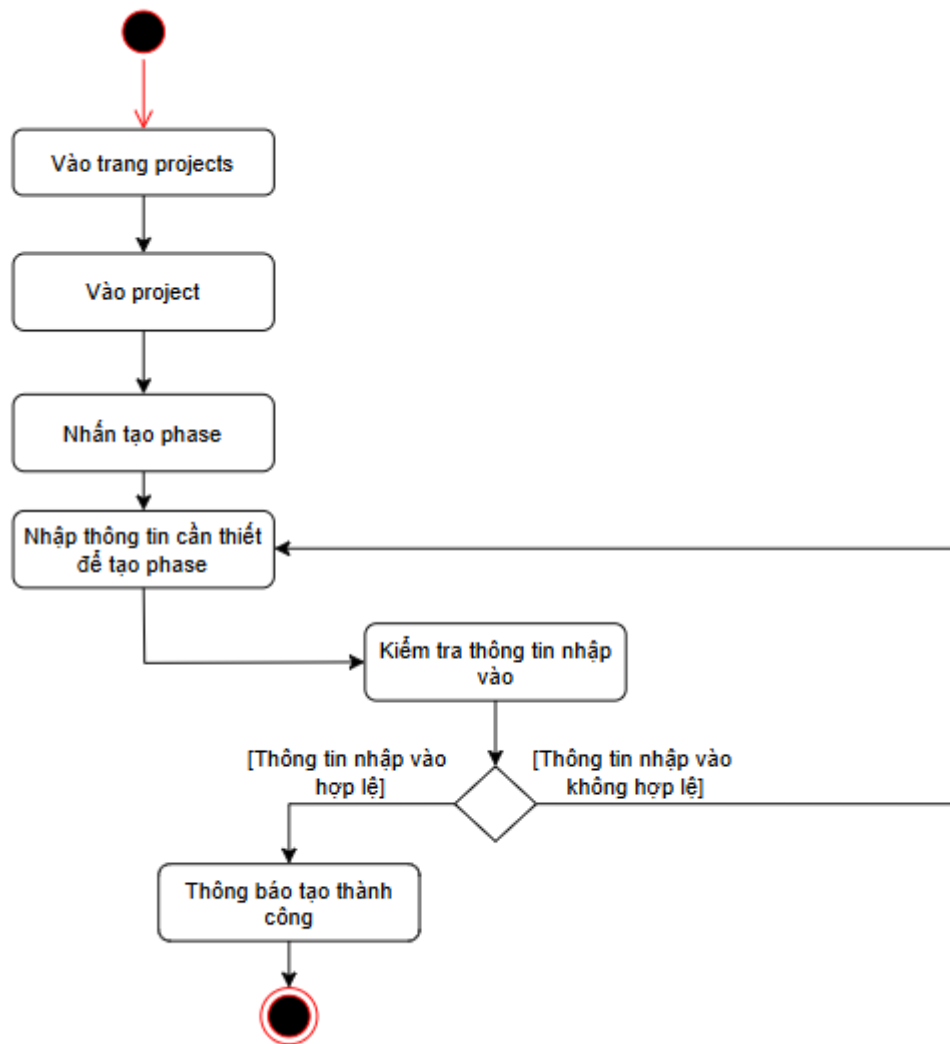
7.8. Ca sử dụng “Thêm thành viên”



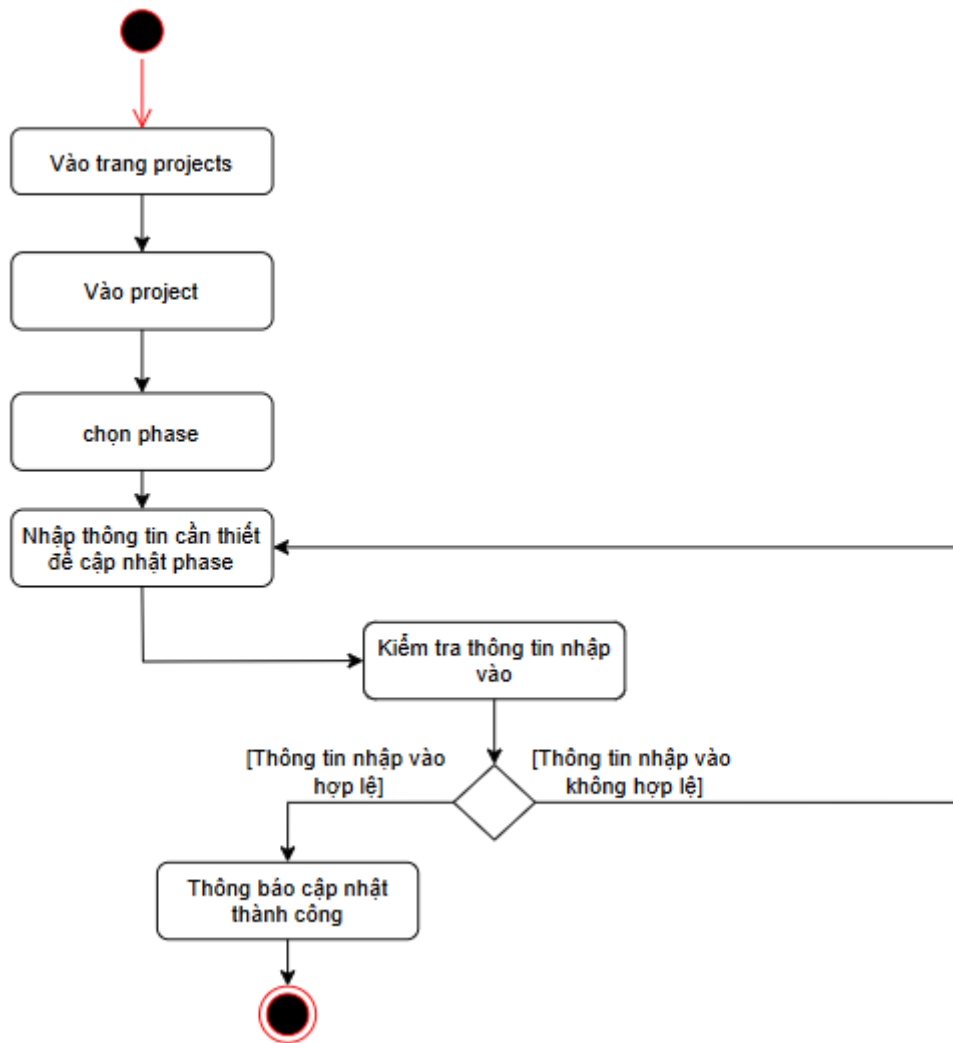
### 7.9. Ca sử dụng “Phân quyền”



7.10. Ca sử dụng “Tạo phase”

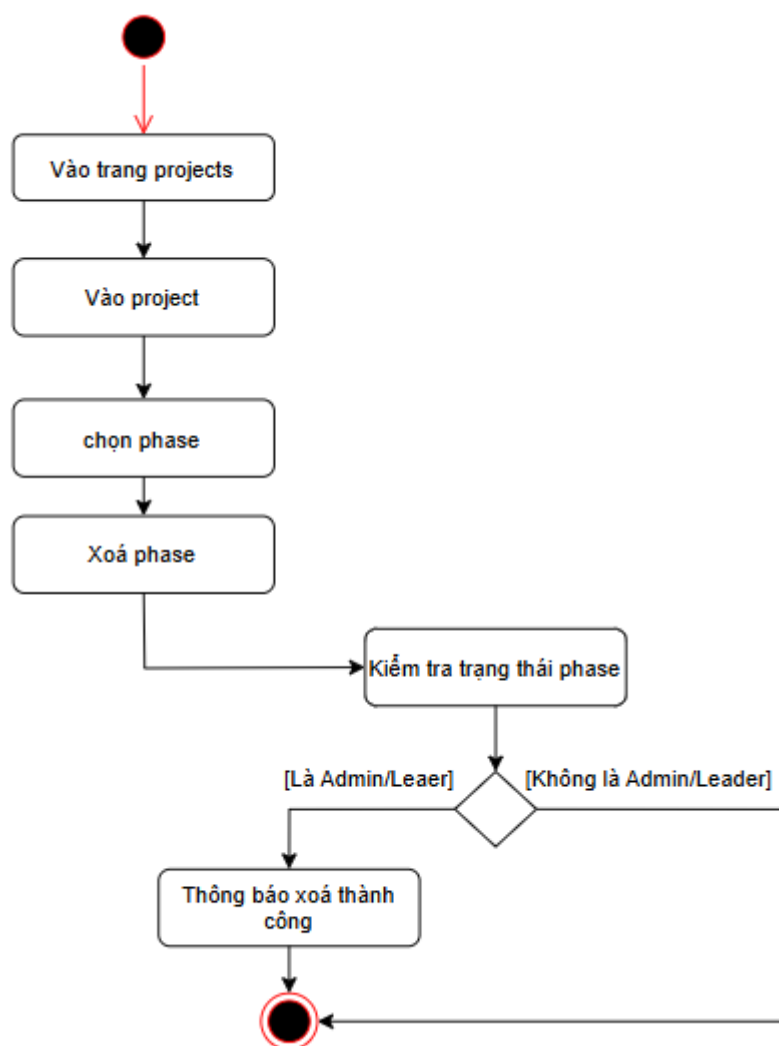


7.11. Ca sử dụng “Cập nhật phase”

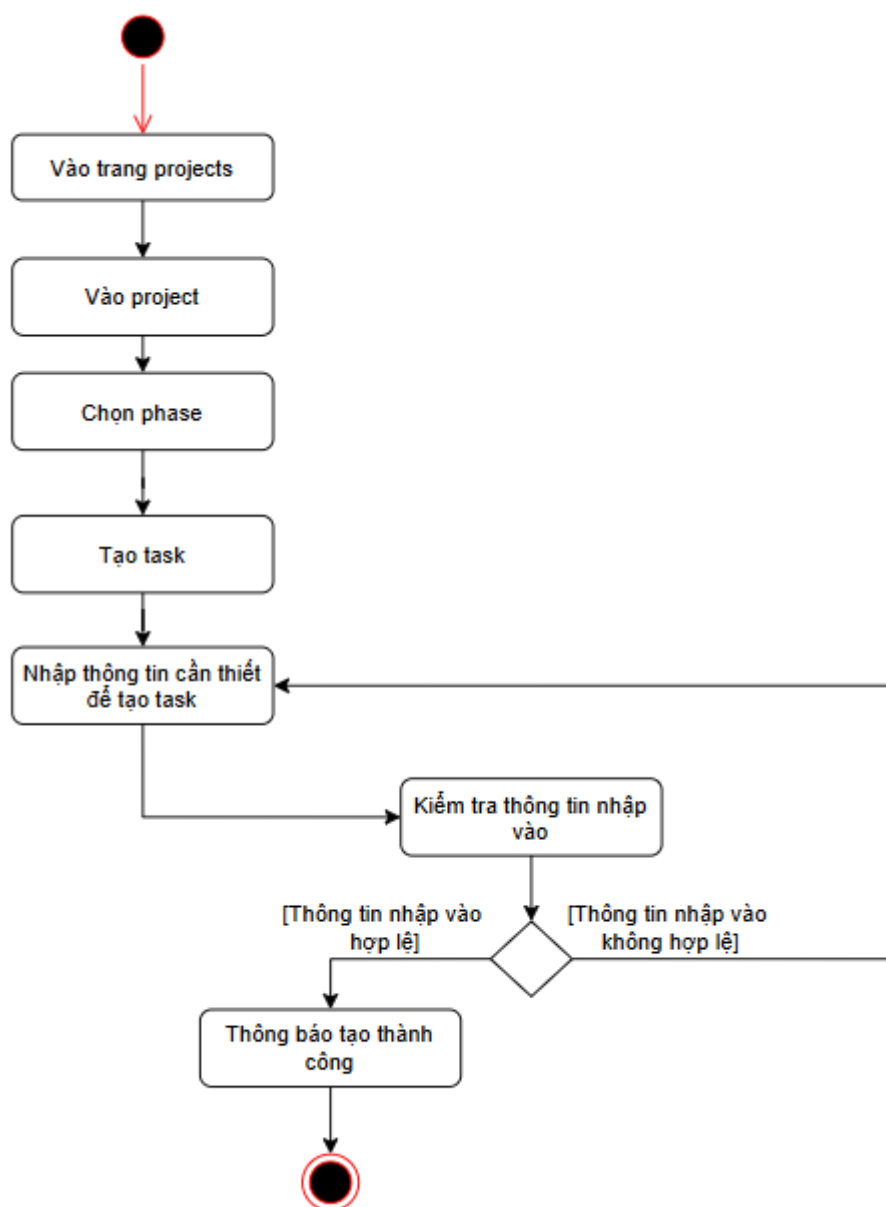




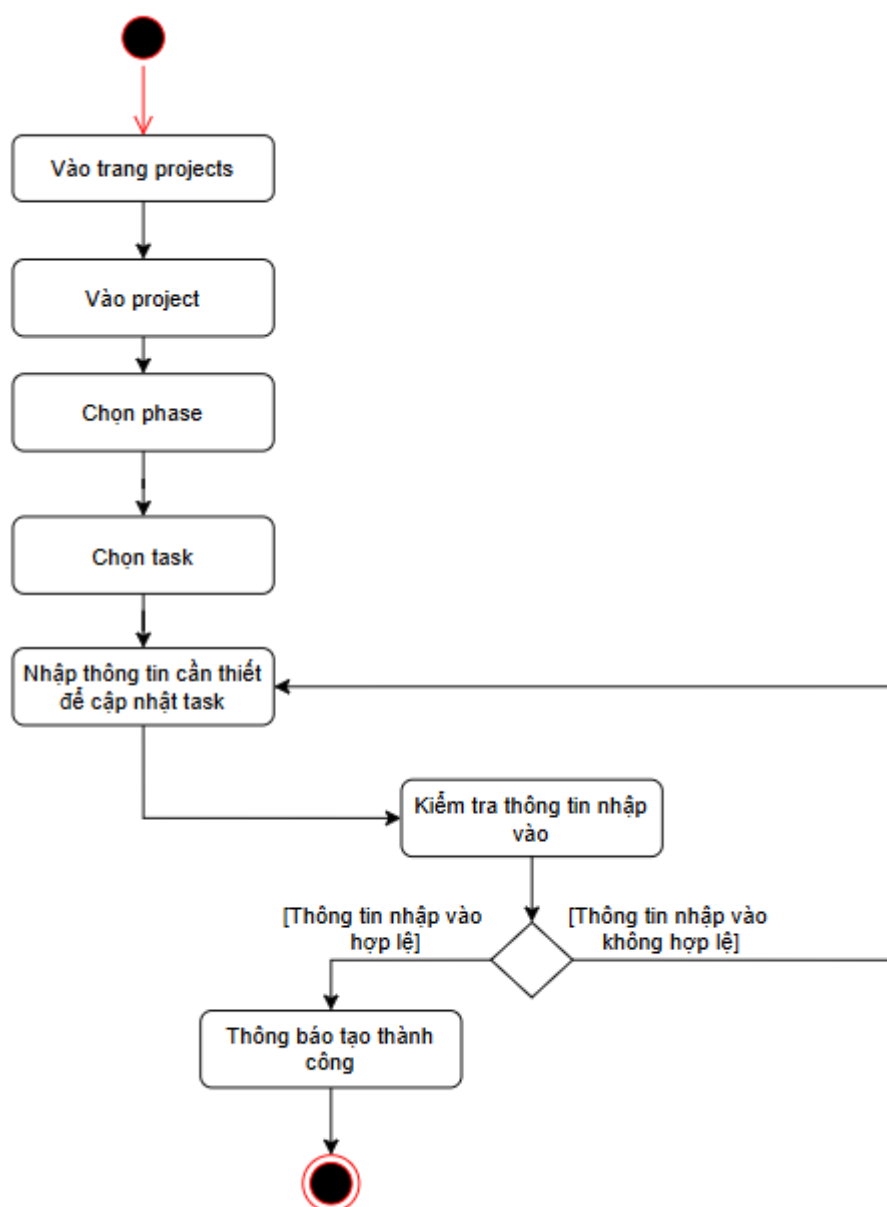
7.12. Ca sử dụng “Xoá phase”



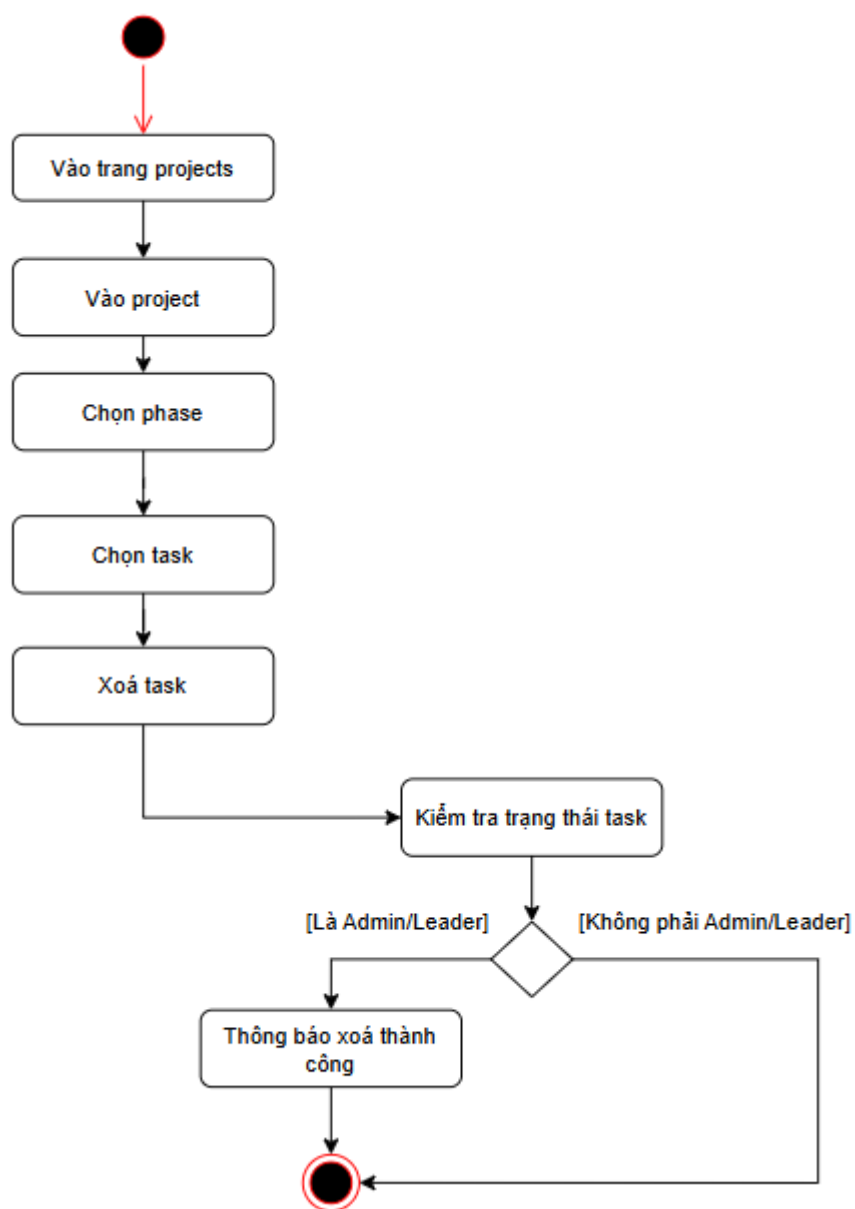
### 7.13. Ca sử dụng “Tạo task”



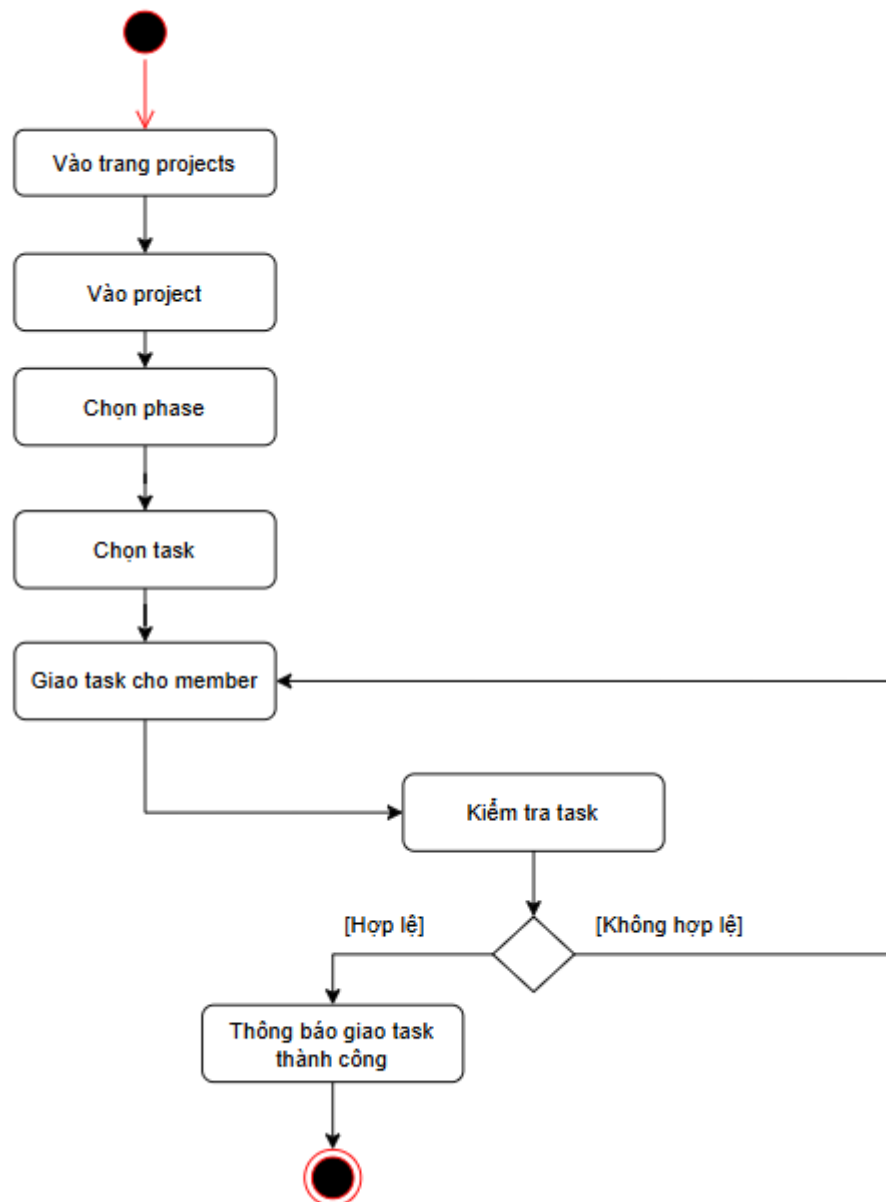
7.14. Ca sử dụng “Cập nhật task”



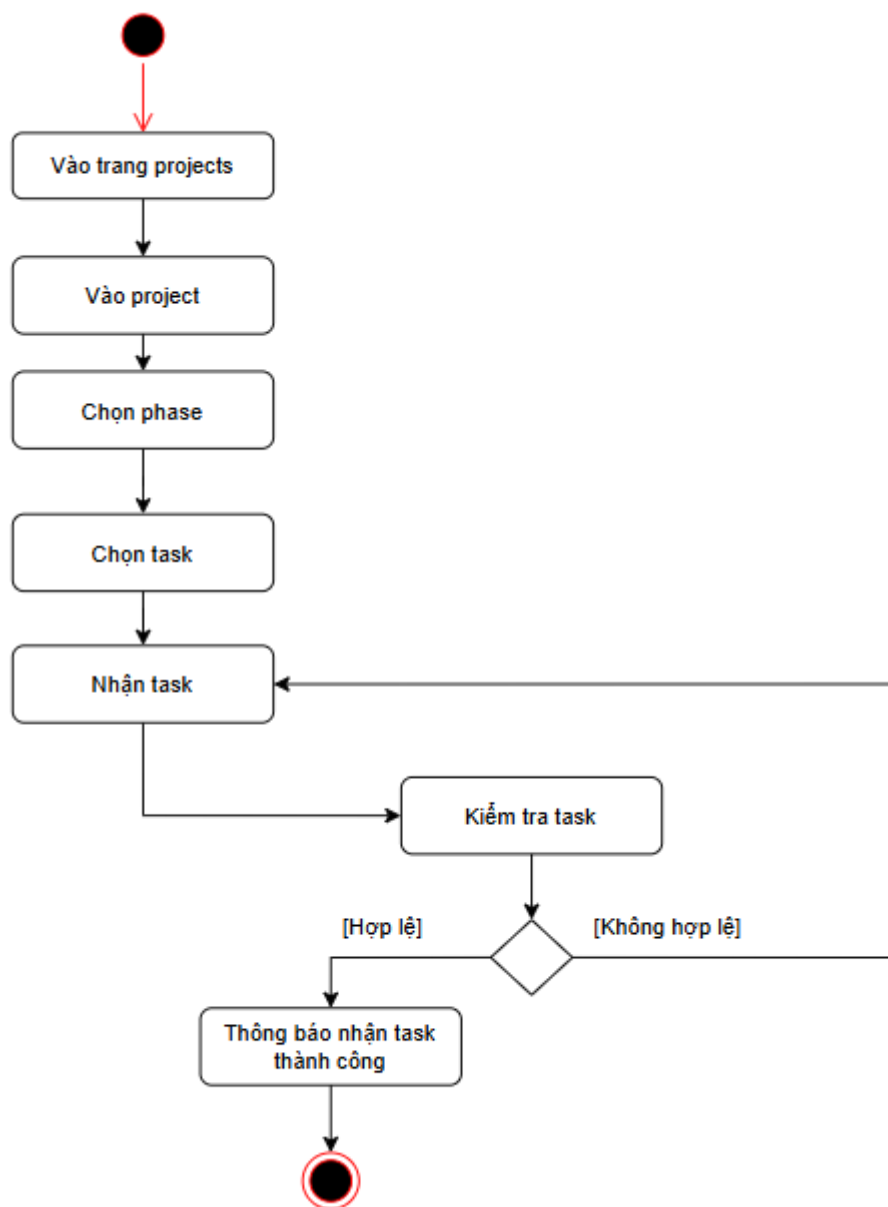
### 7.13. Ca sử dụng “Xóa task”



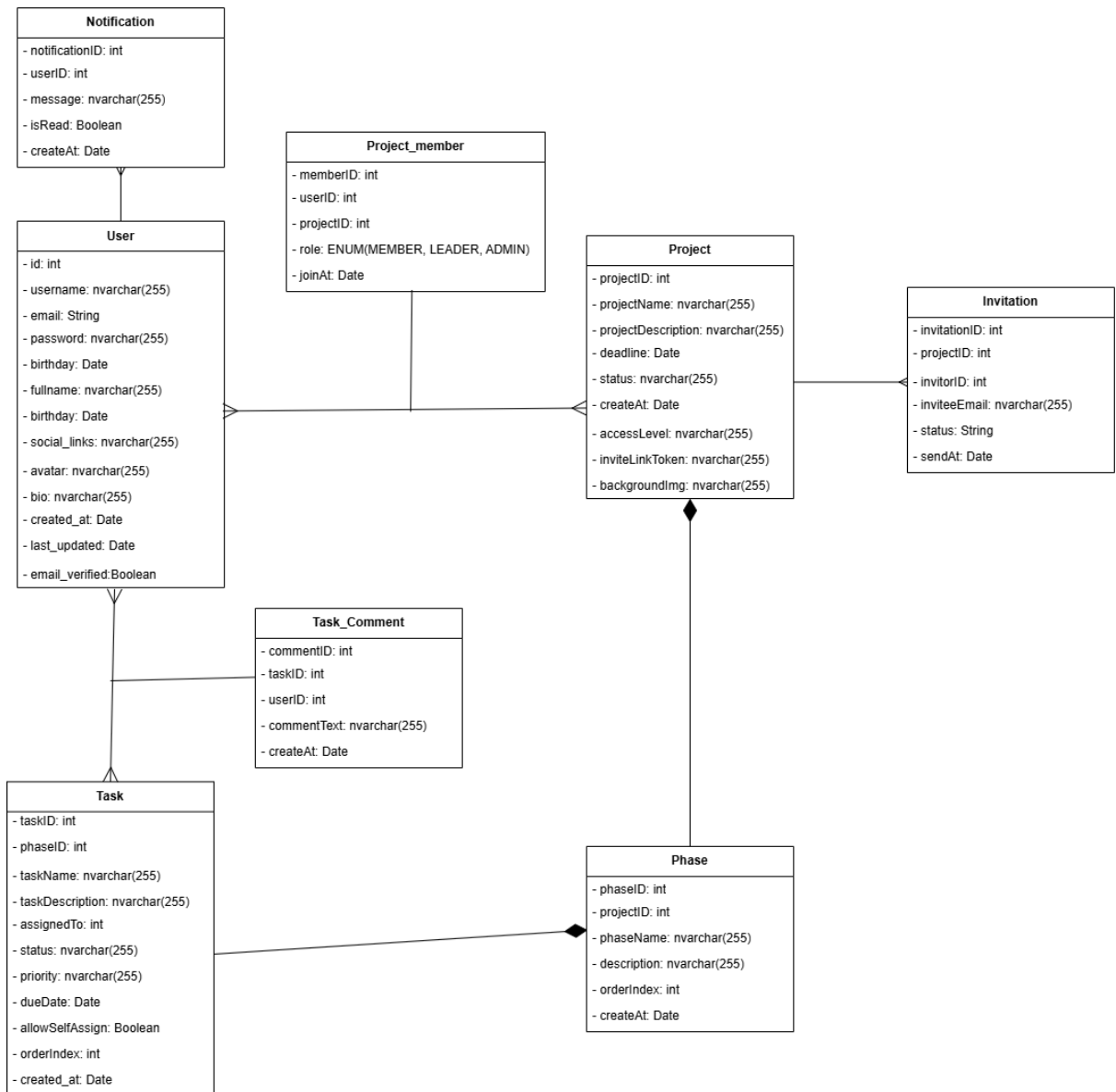
7.13. Ca sử dụng “Giao task”



7.14. Ca sử dụng “Nhận task”



## 7. Sơ đồ lớp



## II. THỰC HIỆN

### 1. Công nghệ sử dụng:

Hệ thống được xây dựng theo mô hình MVC, trong đó:

**Backend (Spring Boot):** Spring Boot giúp rút ngắn thời gian khởi tạo và cấu hình dự án nhờ cơ chế **auto-configuration**, cho phép “chạy luôn” với máy chủ nhúng (Tomcat, Jetty) mà không cần thiết lập thủ công. Ngoài ra, Spring Boot tích hợp sẵn nhiều **starter** (starter dependencies) để dễ dàng thêm module như JPA, Security, hoặc Web, đồng thời có **Spring Actuator** hỗ trợ giám sát và quản lý ứng dụng. Với cộng đồng rộng lớn và tài liệu phong phú, Spring Boot là lựa chọn lý tưởng để phát triển nhanh, bảo trì dễ dàng và mở rộng linh hoạt.

**Frontend (React + HTML/CSS):** Dựa trên React để xây dựng SPA, các component như Dashboard, TaskList, TaskForm, CommentList được phân chia rõ ràng, kết nối REST API qua Axios. Giao diện HTML semantic phối hợp CSS (đảm bảo responsive, phản hồi nhanh và UX trực quan (navbar, sidebar, badge thông báo, modal tạo/sửa task). React Router bảo vệ route theo role; Axios interceptor tự động gắn JWT vào header.

**Cơ sở dữ liệu (MySQL):** Thiết kế schema chuẩn hóa: bảng users, projects, tasks, comments, attachments, notifications, ...

### 2. Mô hình kiến trúc:

Hiện nay, có nhiều mô hình kiến trúc phần mềm phổ biến (Layered Architecture, Microservices, MVVM, Hexagonal, CQRS, v.v.) giúp tách biệt trách nhiệm và tối ưu hóa khả năng mở rộng. Tuy nhiên, trong đề tài “Hệ thống hỗ trợ giao việc và theo dõi tiến độ công việc nhóm,” chúng tôi chọn mô hình MVC (Model–View–Controller) vì các lý do sau:

- **Tách biệt rõ ràng giữa giao diện (View), xử lý nghiệp vụ (Controller) và dữ liệu (Model):** Điều này giúp nhóm phát triển có thể phân công một cách linh hoạt—nhóm frontend chỉ tập trung vào View, trong khi backend tập trung vào logic nghiệp vụ và truy xuất dữ liệu. Khi requirement thay đổi, chỉ cần chỉnh sửa bên layer tương ứng.
- **Dễ bảo trì và mở rộng:** Khi cần bổ sung tính năng mới (ví dụ thêm module báo cáo, tích hợp API bên ngoài), chỉ phải viết thêm controller hoặc mở rộng service mà không ảnh hưởng đến view cũ.

Tuy nhiên **Tương tác thời gian thực phức tạp hơn:** Đối với các yêu cầu real-time (WebSocket, push notification), MVC kết hợp thêm các giải pháp riêng (Spring WebSocket, SSE), sẽ phức tạp hơn so với những mô hình event-driven thuần túy.



### 3. Các mẫu Design Pattern sử dụng

#### Singleton Design Pattern

- Đảm bảo mỗi lớp chỉ có một instance duy nhất.
- Singleton pattern được áp dụng thông qua các annotation của Spring Framework.
- Các annotation được sử dụng để định nghĩa hoặc cấu hình các bean, và phạm vi Singleton được áp dụng mặc định: @Component, @Service, @Repository, @Controller, @RestController, @Autowired.
- **Ưu điểm:** Tiết kiệm tài nguyên, dễ quản lý trạng thái chung (ví dụ: kết nối DB).
- **Nhược điểm:** Khó test nếu không dùng DI framework, tiềm ẩn vấn đề cạnh tranh đa luồng nếu implement không đúng.

#### Template Method

- Template Method định nghĩa “khung” (skeleton) của một thuật toán trong lớp cha (abstract class), rồi giao cho các lớp con (subclass) thực hiện (override) một số bước cụ thể.
- Template Method Pattern định nghĩa bộ khung của một thuật toán trong một phương thức, hoãn một số bước cho các lớp con.
- Áp dụng template method ở các file trong Repository
- **Ưu điểm:** Tái sử dụng được khung chung của thuật toán, giảm lặp code. Dễ bảo trì, chỉ sửa template ở lớp cha để thay đổi luồng thực thi. Cho phép lớp con tùy biến từng bước cụ thể mà không ảnh hưởng phần còn lại.
- **Nhược điểm:** Sinh nhiều lớp con nếu có nhiều biến thể, gây rối. Phụ thuộc chặt giữa lớp cha và con, thay đổi template có thể ảnh hưởng nhiều lớp con. Ít linh hoạt khi cần chọn biến thể thuật toán tại runtime.

#### Adapter Pattern

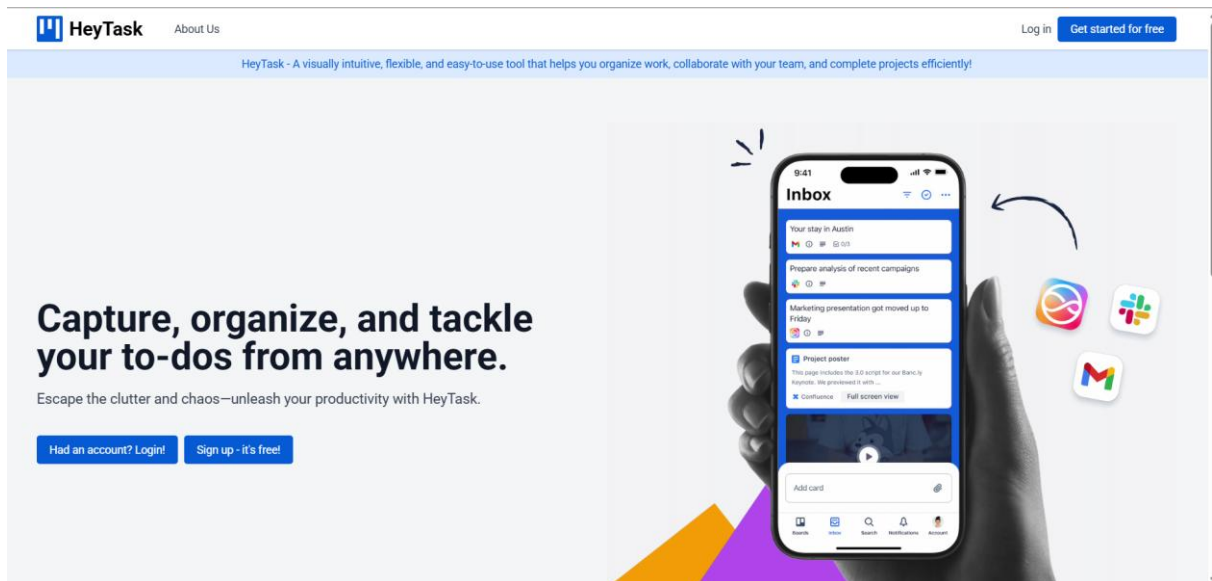
- Adapter Pattern (Mẫu trình điều hợp) là một mẫu cấu trúc (Structural Pattern) cho phép “gói” (wrap) một lớp (hoặc đối tượng) có giao diện không tương thích thành một giao diện mà client yêu cầu. Khi sử dụng, adapter sẽ đóng vai trò trung gian, chuyển đổi (map) các lời gọi của client thành các lời

gọi tương ứng trên lớp gốc, giúp tái sử dụng mã hiện có mà không cần chỉnh sửa class gốc.

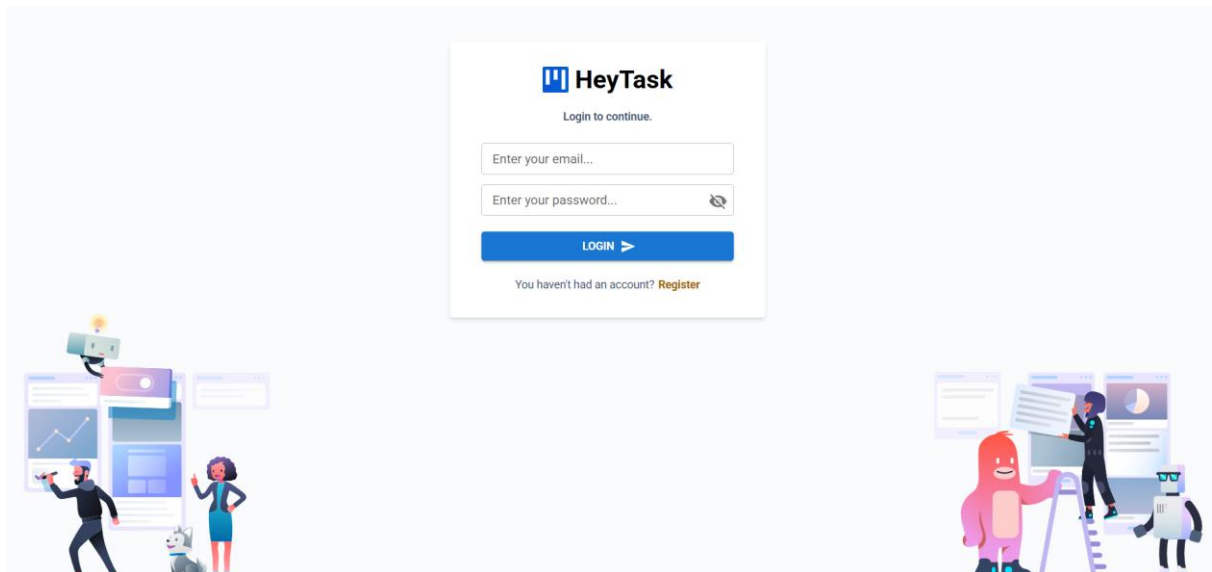
- Adapter Pattern cho phép các interface không tương thích có thể làm việc cùng nhau.
- Ứng dụng adapter pattern ở file UserDetailsImpl.java
- **Ưu điểm:** Cho phép kết nối hai giao diện không tương thích, giúp tái sử dụng mã hiện có mà không cần sửa đổi. Giảm coupling giữa đối tượng khách (client) và lớp dịch (adaptee), vì client chỉ tương tác với adapter. Dễ mở rộng: nếu cần hỗ trợ thêm giao diện mới, chỉ cần viết adapter tương ứng.
- **Nhược điểm:** Tăng số lớp và làm cấu trúc mã phức tạp hơn (phải thêm lớp adapter). Nếu có quá nhiều adapter, sẽ gây khó quản lý và hiểu lầm về luồng xử lý. Đôi khi chỉ đơn giản bọc giao diện để “chuyển đổi” khiến hiệu suất kém hơn so với cách viết trực tiếp.

### III. THIẾT KẾ GIAO DIỆN

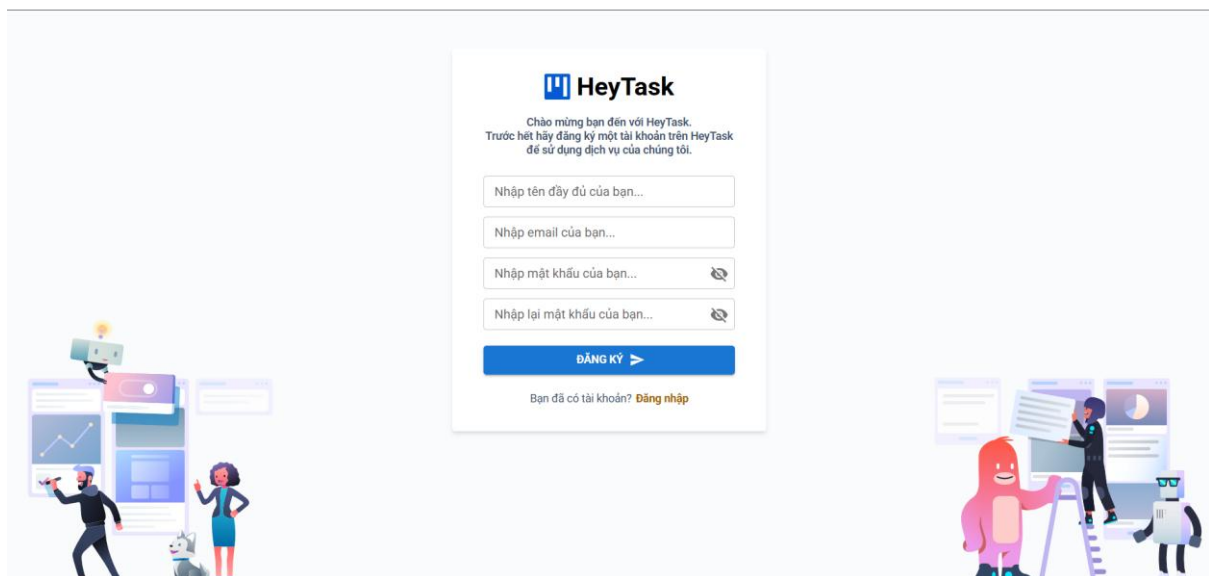
#### 3.1. Giao diện trang chủ



#### 3.2. Giao diện Đăng nhập



### 3.3. Giao diện Đăng ký



The registration interface for HeyTask features a central white card on a light blue background. The card displays the HeyTask logo and a welcome message in Vietnamese. Below the message are four input fields for registration: full name, email, password, and password confirmation. A blue 'ĐĂNG KÝ' (Register) button is positioned below the fields. At the bottom of the card, there is a link for existing users to 'Đăng nhập' (Log in). The background is decorated with colorful illustrations of people and robots interacting with digital screens.

**HeyTask**

Chào mừng bạn đến với HeyTask.  
Trước hết hãy đăng ký một tài khoản trên HeyTask để sử dụng dịch vụ của chúng tôi.

Nhập tên đầy đủ của bạn...

Nhập email của bạn...

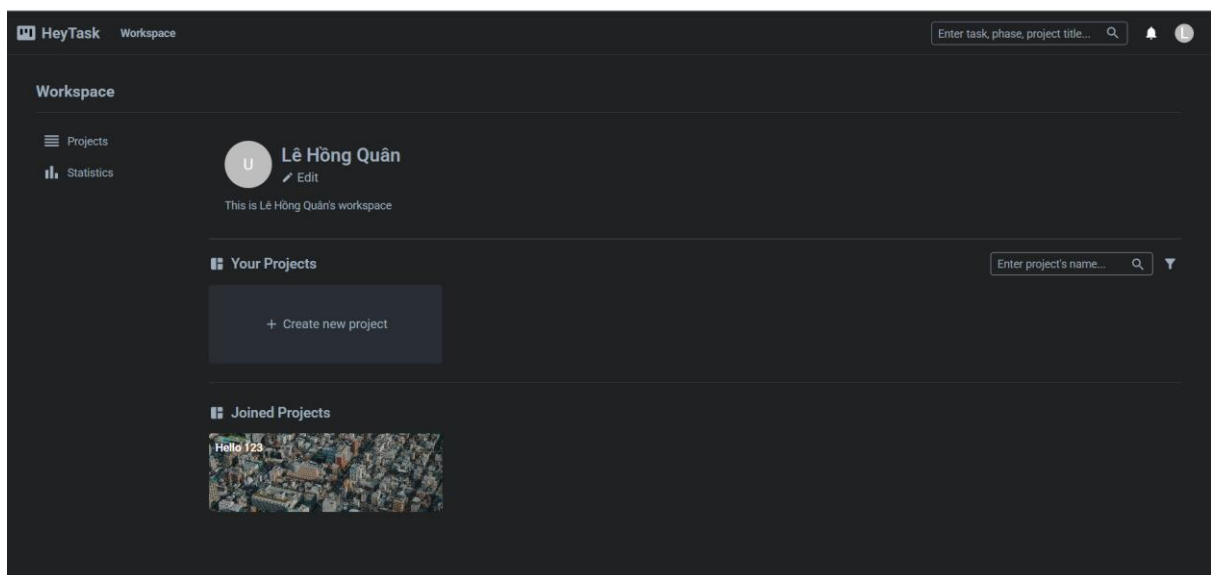
Nhập mật khẩu của bạn...

Nhập lại mật khẩu của bạn...

**ĐĂNG KÝ**

Bạn đã có tài khoản? [Đăng nhập](#)

### 3.4. Giao diện Workspace



The workspace interface for HeyTask has a dark theme. At the top, there is a header bar with the HeyTask logo, the word 'Workspace', and a search bar. Below the header, the main area is divided into sections. On the left, there is a sidebar with 'Projects' and 'Statistics' options. The main content area shows the user's profile, 'Lê Hồng Quân', with an 'Edit' link. Below the profile, there is a section for 'Your Projects' with a '+ Create new project' button. At the bottom, there is a section for 'Joined Projects' with a preview image and the text 'Hello 123'.

**HeyTask** Workspace

Enter task, phase, project title...

**Workspace**

Projects  
Statistics

**Lê Hồng Quân**  
Edit

This is Lê Hồng Quân's workspace

**Your Projects**

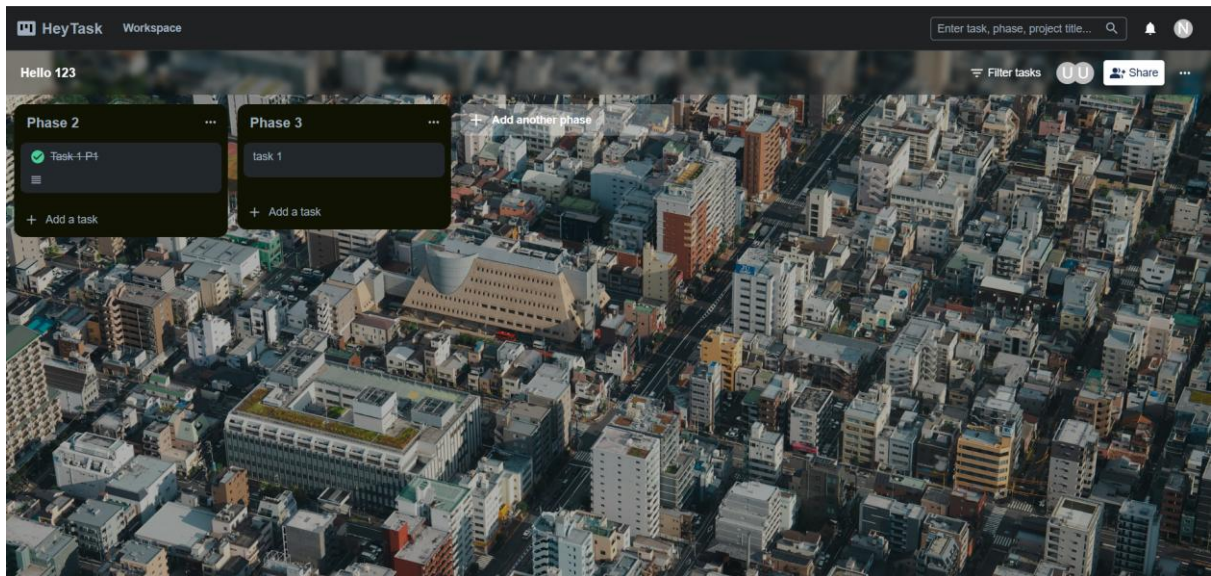
Enter project's name...

+ Create new project

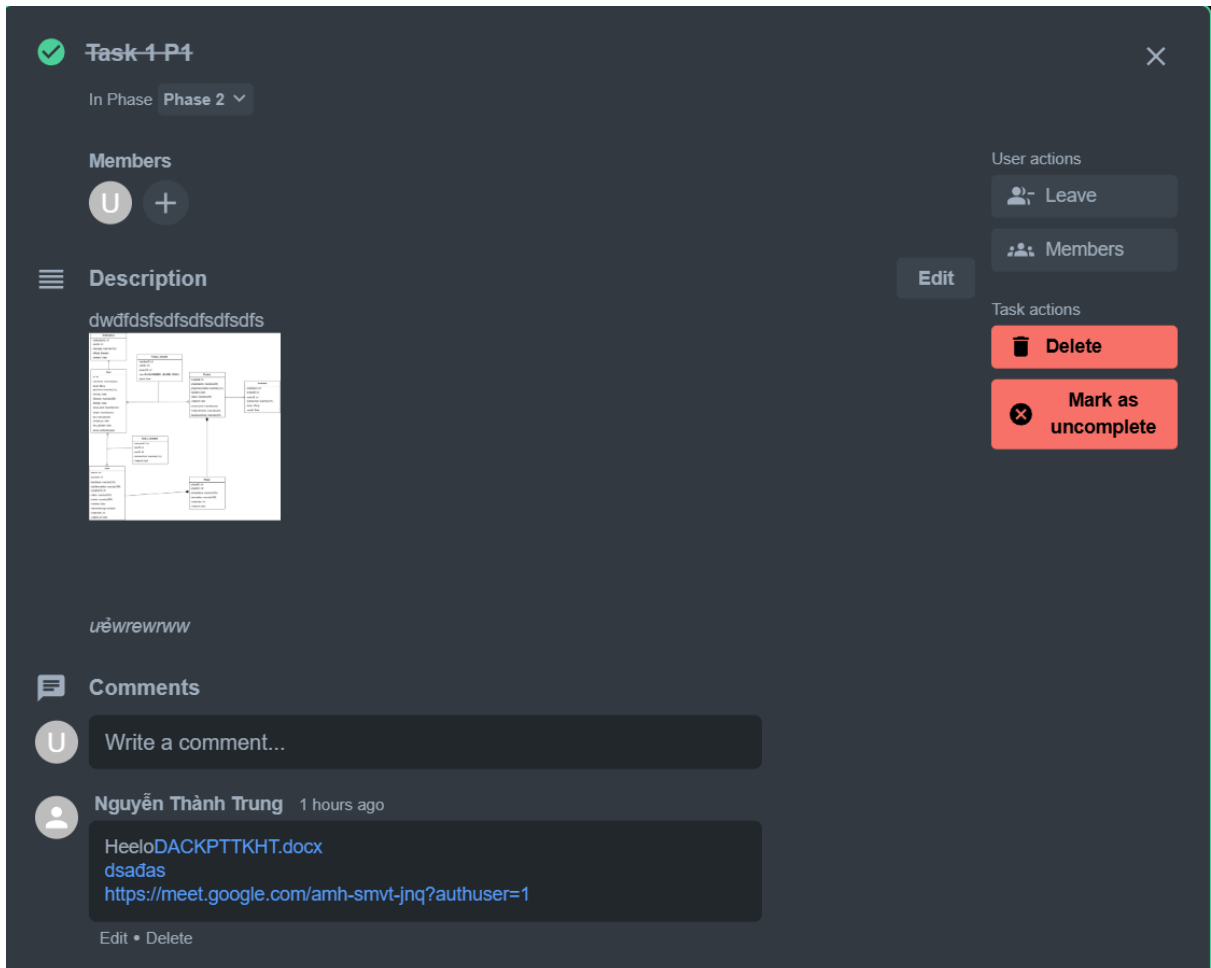
**Joined Projects**

Hello 123

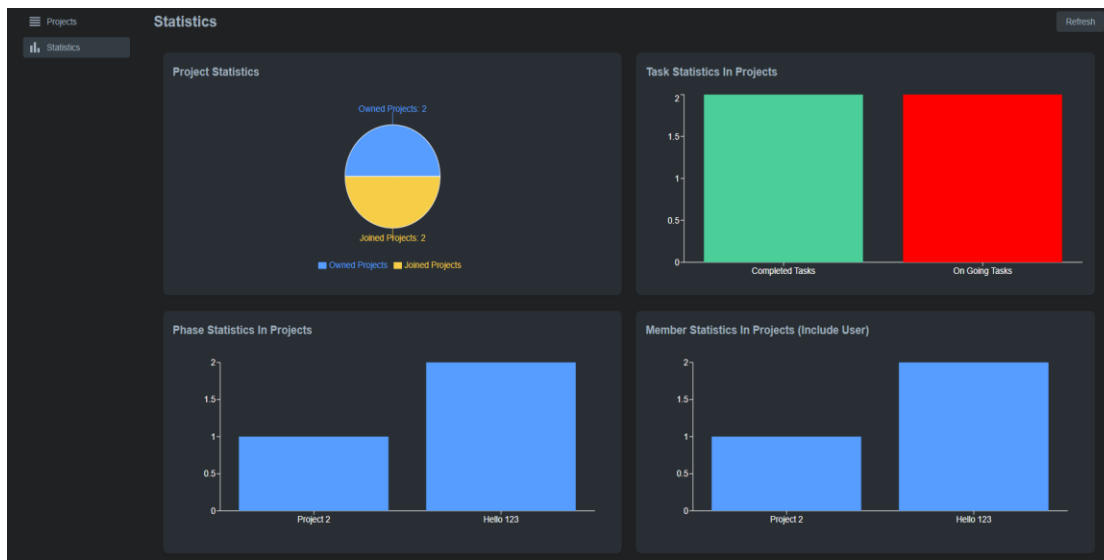
### 3.5. Giao diện Project



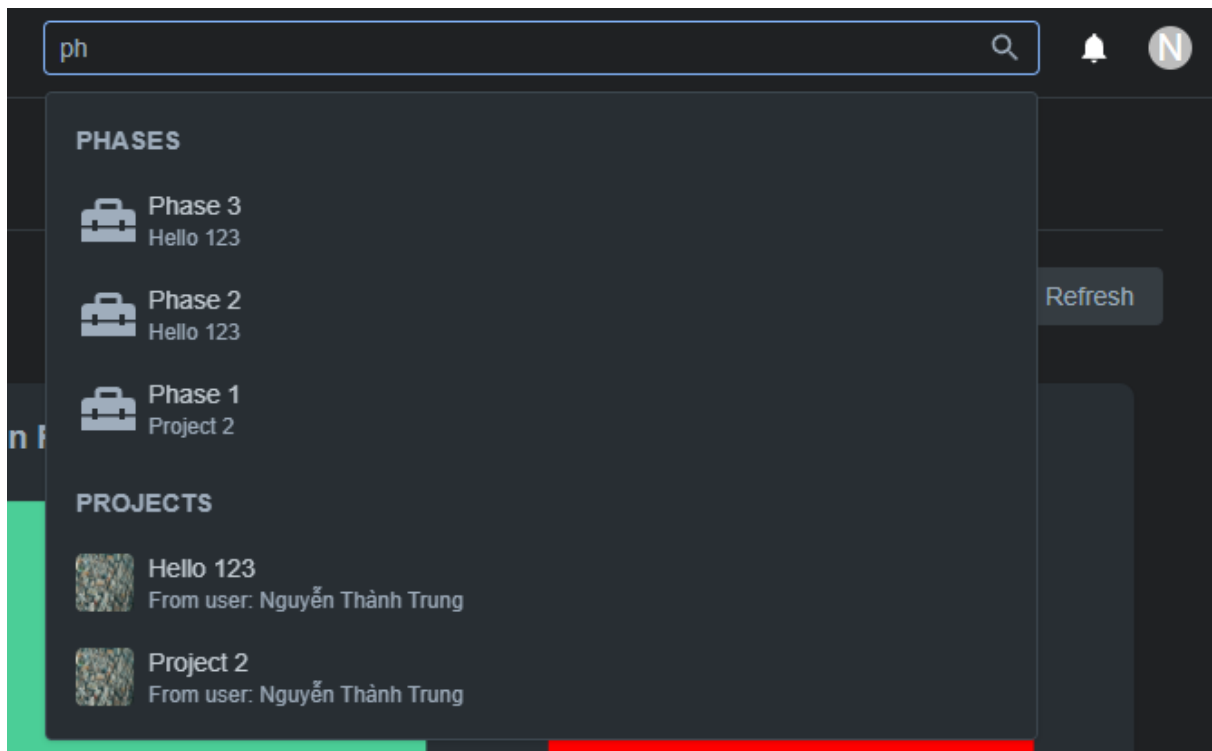
### 3.6. Giao diện Task:



### 3.7. Giao diện Thống kê




### 3.8. Giao diện tìm kiếm



### 3.9. Giao diện tài khoản

**Profile photo and header image**



**About you**

Full name

Nguyễn Thành Trung

Email (Read only)

nguyenthant632@gmail.coi

Bio

<https://github.com/tuan03?tab=repositories>

Gender

Female

Birthday


2003-06-09

Social link


<https://github.com/tuan03?tab=repositories>

**Password**

Current password

Enter your full name here... 

New password

Enter your full name here... 

## VI. KẾT LUẬN

Đề tài “Xây dựng hệ thống hỗ trợ giao việc và theo dõi tiến độ công việc nhóm” đã triển khai thành công giải pháp trực tuyến, giúp khắc phục hầu hết những hạn chế của quy trình giao việc truyền thống như thông tin phân tán, khó đồng bộ và thiếu khả năng theo dõi thời gian thực. Hệ thống được phát triển trên nền tảng MVC kết hợp Layered Architecture, đảm bảo tách bạch rõ ràng giữa giao diện, xử lý nghiệp vụ và truy cập dữ liệu. Việc ứng dụng đồng bộ các Design Patterns (Singleton, Factory, Builder, Strategy, Chain of Responsibility, Proxy, Observer, Command, Adapter, Facade, Decorator) đã giúp nâng cao tính linh hoạt, khả năng mở rộng và bảo trì của hệ thống.

Trong quá trình triển khai, nhóm phát triển đã thực hiện đầy đủ các bước: khảo sát thực tế, thu thập và phân tích yêu cầu, thiết kế UML (Use Case, Class, Sequence, Activity), xây dựng cơ sở dữ liệu SQL Server chuẩn hóa, cũng như thiết kế giao diện người dùng thân thiện, hỗ trợ responsive. Kết quả đầu ra gồm các chức năng chính: tạo/sửa/xóa task, phân quyền rõ ràng giữa quản lý và thành viên, cập nhật trạng thái công việc theo thời gian thực, comment & đính kèm tài liệu, hệ thống cảnh báo và gửi thông báo tự động khi task thay đổi hoặc quá hạn, cùng tính năng xuất báo cáo tiến độ và khối lượng công việc theo tuần/tháng.

Hệ thống đã được kiểm thử cơ bản và vận hành ổn định trong môi trường giả lập, đáp ứng yêu cầu về hiệu năng khi số lượng task và người dùng ở mức vừa phải. Giao diện UX thân thiện giúp người dùng (quản lý, thành viên) dễ dàng tương tác, giảm thiểu thời gian làm quen. Cơ sở dữ liệu chuẩn hóa và kiến trúc phân lớp cho phép mở rộng thêm các tính năng.

Tóm lại, đề tài đã đạt được mục tiêu đề ra: xây dựng một công cụ tập trung, minh bạch và linh hoạt để giao việc và theo dõi tiến độ công việc nhóm. Hệ thống không chỉ cải thiện hiệu quả phối hợp giữa các thành viên mà còn hỗ trợ quản lý ra quyết định kịp thời dựa trên dữ liệu thời gian thực. Về hướng phát triển tiếp theo, có thể cân nhắc các nâng cấp sau: tối ưu hiệu năng khi số lượng task/nhiệm vụ lớn, bổ sung phân tích dữ liệu dựa trên Machine Learning để dự báo tiến độ, và tích hợp thêm các kênh thông báo đa dạng (mobile push notification, chatbots). Với những nền tảng và thiết kế đã hoàn thiện, hệ thống có đủ khả năng mở rộng, đáp ứng tốt các yêu cầu phức tạp hơn trong tương lai.