

**TÀI LIỆU MÔ TẢ THIẾT KẾ PHẦN MỀM**

***Hệ thống quản lý Dự án***

**Hà Nội, [tháng/năm]**

**Bảng ghi nhận sự thay đổi của tài liệu**

| **Thời gian thay đổi** | **Nội dung thay đổi** | **Lý do** | **Phiên bản bị thay đổi** | **Mô tả sự thay đổi** | **Phiên bản mới** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

MỤC LỤC

[1. GIỚI THIỆU 5](#_Toc322290205)

[1.1. Mục đích tài liệu 5](#_Toc322290206)

[1.2. Phạm vi tài liệu 5](#_Toc322290207)

[1.3. Thuật ngữ và các từ viết tắt 5](#_Toc322290208)

[1.4. Tài liệu tham khảo 5](#_Toc322290209)

[1.5. Mô tả tài liệu 5](#_Toc322290210)

[2. TỔNG QUAN VỀ PHẦN MỀM 6](#_Toc322290211)

[3. THIẾT KẾ KIẾN TRÚC PHẦN MỀM 7](#_Toc322290212)

[3.1. Mô hình kiến trúc 7](#_Toc322290213)

[3.2. Mô tả kiến trúc 7](#_Toc322290214)

[4. THIẾT KẾ DỮ LIỆU 8](#_Toc322290215)

[4.1. Mô tả dữ liệu 8](#_Toc322290216)

[4.2. Từ điển dữ liệu 8](#_Toc322290217)

[5. THIẾT KẾ CÁC THÀNH PHẦN (PHÂN HỆ) 9](#_Toc322290218)

[5.1. Thành phần 1 9](#_Toc322290219)

[5.2. Thành phần 2 9](#_Toc322290220)

[… 9](#_Toc322290221)

[5.n. Thành phần n 9](#_Toc322290222)

[6. THIẾT KẾ GIAO DIỆN NGƯỜI SỬ DỤNG 9](#_Toc322290223)

[6.1. Mô tả tổng quan 9](#_Toc322290224)

[6.2. Hình ảnh giao diện 9](#_Toc322290225)

[6.3. Các đối tượng giao diện và hoạt động đi kèm 9](#_Toc322290226)

[7. PHỤ LỤC 10](#_Toc322290227)

# 1. GIỚI THIỆU

## 1.1. Mục đích tài liệu

Đây là tài liệu thiết kế phần mềm cho hệ thống quản lý dự án. Tài liệu này mô tả chi tiết các tính năng, cấu trúc dữ liệu và quy trình hoạt động bên trong của phần mềm. Nó là nền tảng quan trọng để các lập trình viên thực hiện quá trình phát triển phần mềm.

## 1.2. Phạm vi tài liệu

Phạm vi của tài liệu thiết kế phần mềm bao gồm tất cả các thông tin chi tiết cần thiết để mô tả đầy đủ một hệ thống phần mềm, từ khái niệm ban đầu cho đến khi hoàn thiện, thể hiện sự tương tác giữa các thành phần và các usecase của hệ thống. Ngoài ra tài liệu còn là bản thiết kế về sơ đồ lớp, sơ đồ tuần tự của từng phân hệ và usecase. Tài liệu này đóng vai trò như một bản thiết kế chi tiết, hướng dẫn các nhà phát triển thực hiện quá trình xây dựng phần mềm một cách chính xác và hiệu quả.

## 1.3. Thuật ngữ và các từ viết tắt

|  |  |
| --- | --- |
| Từ ngữ | Định nghĩa |
| SRS | Software Requirements Specification – đặc tả yêu cầu phần mềm |
| Milestones | Các cột mốc về kế hoạch |
| Releases | Các cột mốt bản bàn giao cho dự án |
| Issues | Các vấn đề phát sinh trong dự án |
| Members | Các thành viên trong đội ngũ phát triển, quản lý dự án |
| Department | Tên gọi chung cho một phòng ban/trung tâm |

## 1.4. Tài liệu tham khảo

* **GitLab:** Một nền tảng quản lý dự án phổ biến, được sử dụng làm tham khảo để xây dựng các tính năng quản lý dự án, milestones, releases và issues.

## 1.5. Mô tả tài liệu

Tài liệu này có cấu trúc gồm 3 phần chính:

Tổng quản về phần mềm: mô tả khác quát về yêu cầu của hệ thống quản lý dự án.

Thiết kế kiến trúc phần mềm: đưa ra mô hình kiến trúc và mô tả kiến trúc thông qua các sơ đồ uml: sơ đồ ngữ cảnh, sơ đồ đối tượng và sơ đồ tuần tự.

Thiết kế dữ liệu: đưa ra mô tả dữ liệu và từ điển dữ liệu. Phần này sẽ mô tả, chỉ rõ việc hệ thống hóa các đối tượng trong yêu cầu thành các dạng đối tượng trong dữ liệu.

# 2. TỔNG QUAN VỀ PHẦN MỀM

2.1 Yêu cầu của phần mềm:

Phát triển hệ thống quản lý dự án chuyên nghiệp, tích hợp các chức năng quản lý issues, milestones, releases, và nhân sự cho môi trường doanh nghiệp với tính năng phân quyền linh hoạt.

2.2 Môi trường triển khai

- Nền tảng: Máy chủ Linux (Ubuntu Server 22.04LTS)

- Kiến trúc: Client - Server

2.3 Công cụ/công nghệ sử dụng

- Backend (Server)

* Ngôn ngữ lập trình: Java 17
* Framework: Spring Boot, Spring Security, Spring Data
* ORM: Hibernate

Dạng truy vấn: REST API

- Frontend (Client)

* Ngôn ngữ lập trình: javascript
* Framework: React 18
* UI Library: UI-Material

- Cơ sở dữ liệu

* PostgreSQL 14

2.4 Phương pháp thiết kế

- Clean Architecture: Tách biệt các lớp thành phần

- Domain Driven Design: Tập trung vào logic của từng thành phần

- SOLID Principle: Đảm bảo tính mở rộng và bảo trì

2.5 Ràng buộc thiết kế

- Ràng buộc về hiệu năng:

* Thời gian phản hồi tối đa: 300ms
* Khả năng chịu tải: 500 người dùng cùng 1 lúc
* Uptime: 99%

- Ràng buộc về bảo mật:

* Xác thực: sử dụng Oauth2
* Mã hóa: sử dụng SSL/TLS

- Ràng buộc về giao diện:

* Responsive: Hỗ trợ nhiều loại thiết bị có kích cỡ màn hình khác nhau
* Trình duyệt: Chrome, Safari, Firefox
* Ngôn ngữ: Tiếng Việt

# 3. THIẾT KẾ KIẾN TRÚC PHẦN MỀM

## 3.1. Mô hình kiến trúc

***3.1.1 Các thành phần của phần mềm:***

- Quản lý người dùng:

* Quản lý thông tin nhân sự.
* Phân quyền truy cập.
* Xác thực nhân sự.

- Quản lý phòng ban/trung tâm:

* Tạo và lưu trữ thông tin dự án.
* Triển khai các dự án.
* Phân bổ nhân viên và quyền truy cập dự án.

- Quản lý dự án.

* Tạo và lưu trữ thông tin dự án.
* Quản lý nhân sự dự án.

- Quản lý milestones.

* Tạo và lưu trữ thông tin milestones.
* Liên kết với dự án kế hoạch phát triển.

- Quản lý releases.

* Tạo và lưu trữ thông tin releases.
* Liên kết với dự án, milestones và thời gian phát hành các phiên bản dự án.

- Quản lý issues.

* Tạo và theo dõi issues.
* Phân công nhân viên công việc.
* Quản lý trạng thái issue.
* Liên kết với các mốc milestones và releases của dự án.

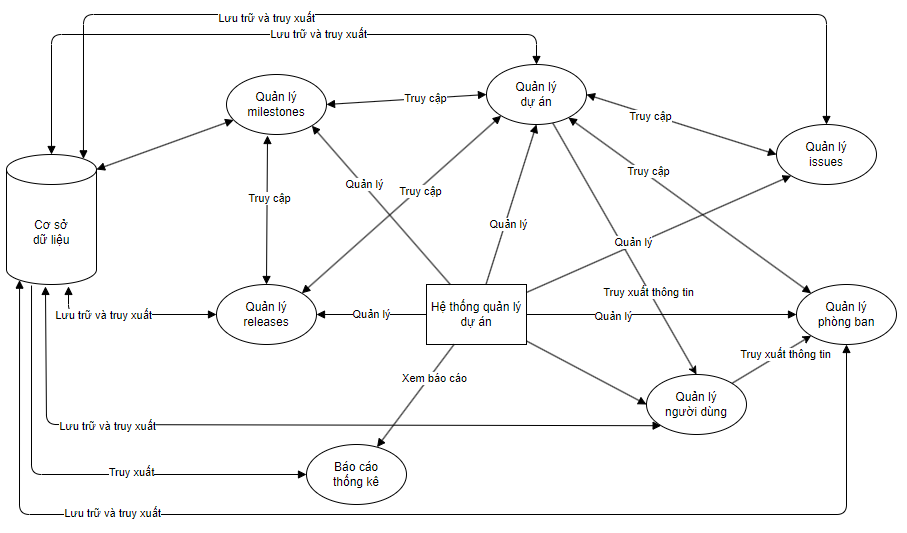
- Báo cáo, thống kê.

* Tạo báo cáo thống kê.
* Xuất báo cáo.

- Cơ sở dữ liệu:

* Lưu trữ, truy xuất dữ liệu.

***3.1.2 Sơ đồ ngữ cảnh:***

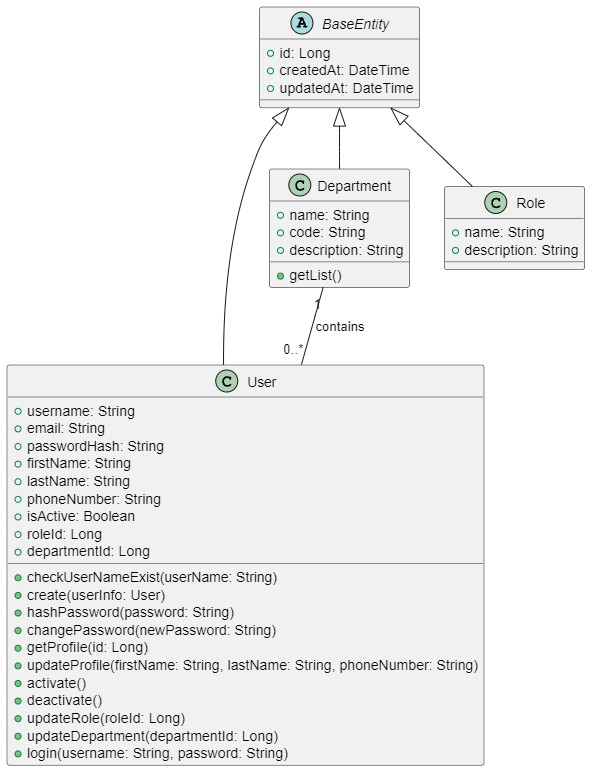


## 3.2. Mô tả kiến trúc

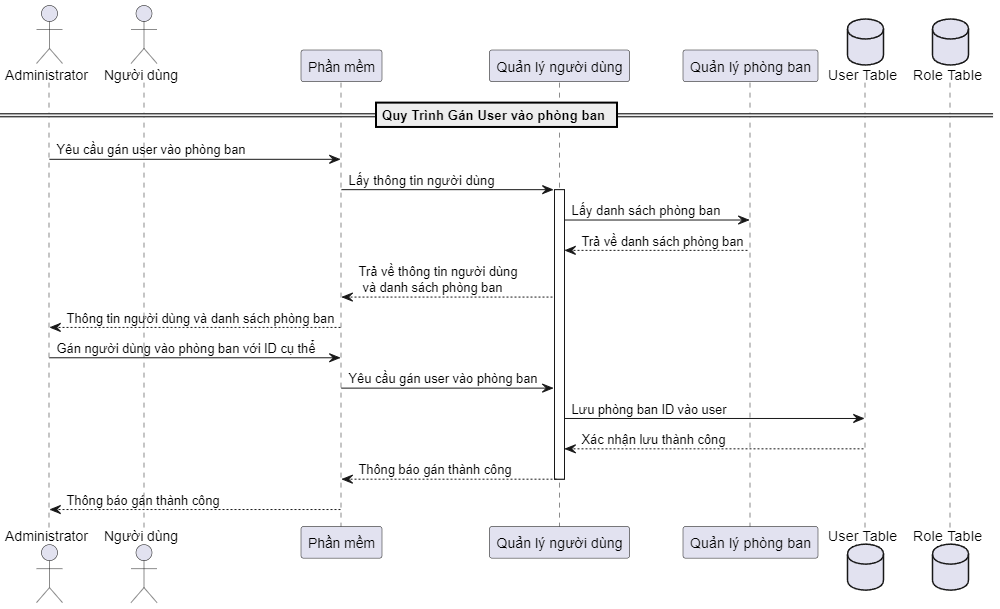
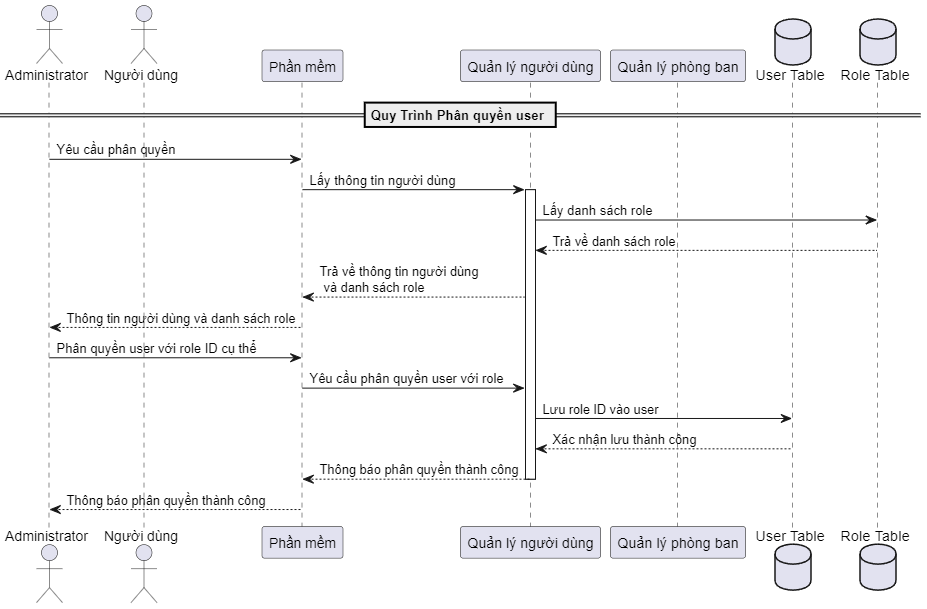
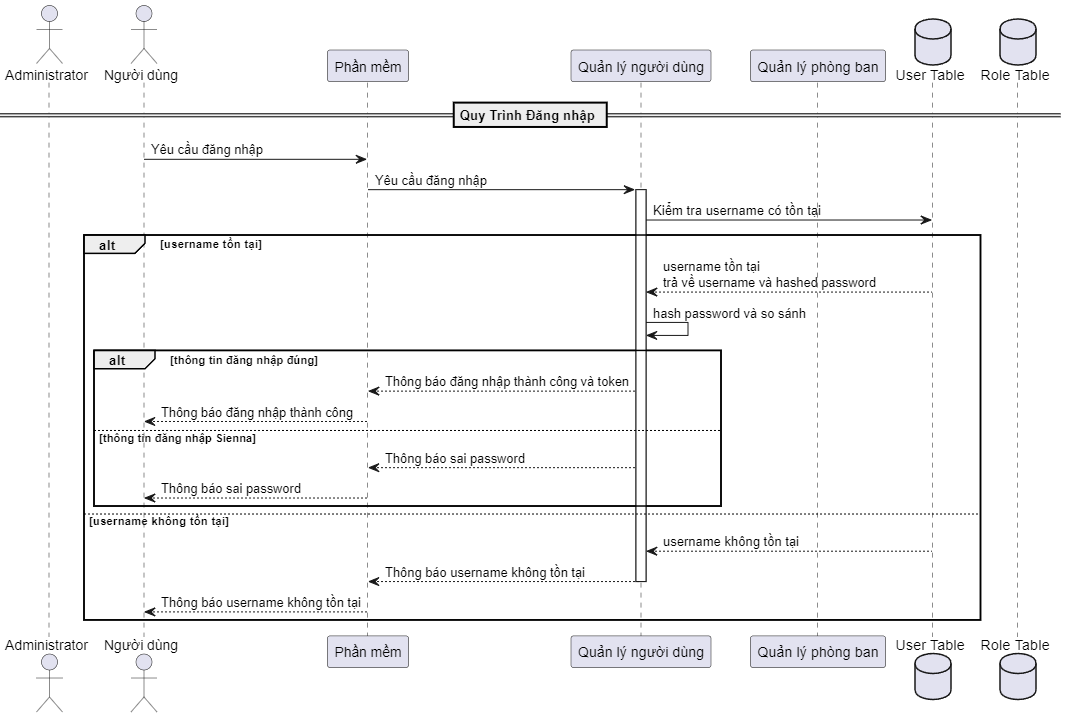
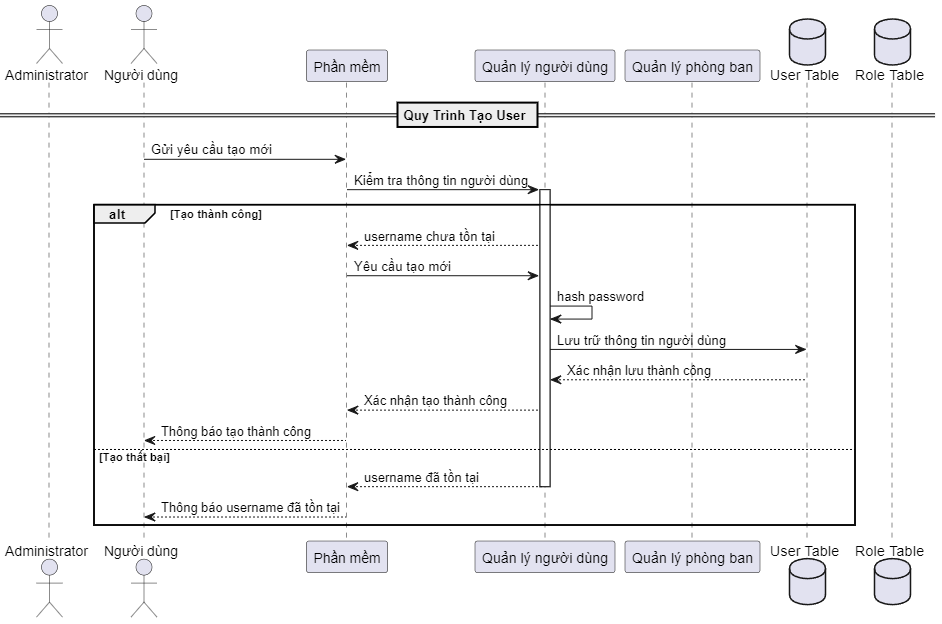
***3.2.1 Quản lý người dùng:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp User: chứa thông tin user và các phương thức để tương tác giữa User và các lớp khác. Các phương thức:
  + checkUserNameExist(username: string): nhận vào tên 1 username và kiểm tra xem username có tồn tại chưa. Phục vụ usecase đăng ký user.
  + create(UserInfo: User): thực hiện việc tạo thông tin 1 user đã được xác thực xuống database.
  + hashPassword(password: string): thực hiện biến đổi plain password thành dạng hash để lưu trữ xuống database.
  + changePassword(newPassword: string): thực hiện việc cập nhật 1 pasword mới xuống database.
  + getProfile(id: Long): lấy thông tin của 1 user bằng userId.
  + updateProfile(firstName: string, lastName: string): thực hiện việc cập nhật firstname và lastname mới cho user.
  + activate(): thực hiện cập nhật trạng thái active cho user.
  + deactivate(): thực hiện việc cập nhật trạng thái deactive cho user.
  + updateRole(roleId: Long): thực hiện việc cập nhật role mới cho user.
  + updateDepartment(departmentId: Long): thực hiện việc cập nhật phòng ban mới cho user.
  + login(username: string, password: string): thực hiện việc đăng nhập của user.
* Lớp Role: chứa thông tin về các role của người dùng.
* Lớp Department: thuộc phân hệ quản lý phòng ban, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa người dùng và phòng ban.



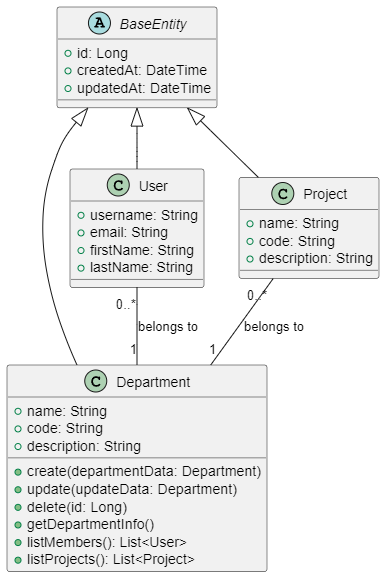
- Sơ đồ tuần tự:



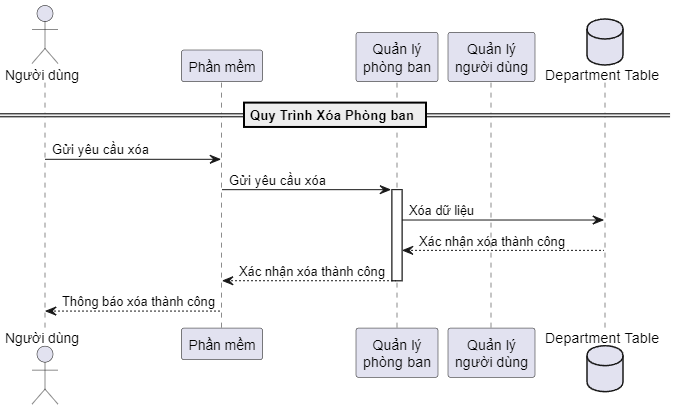
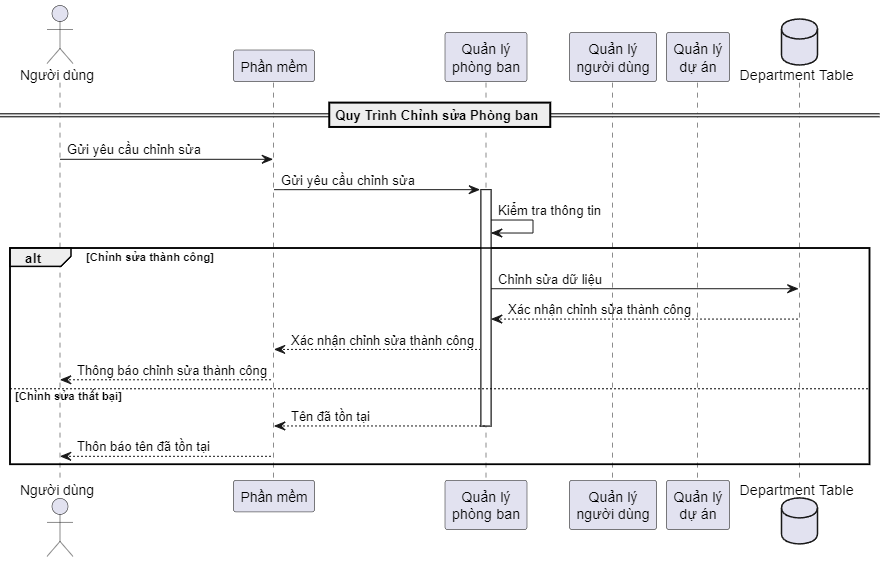
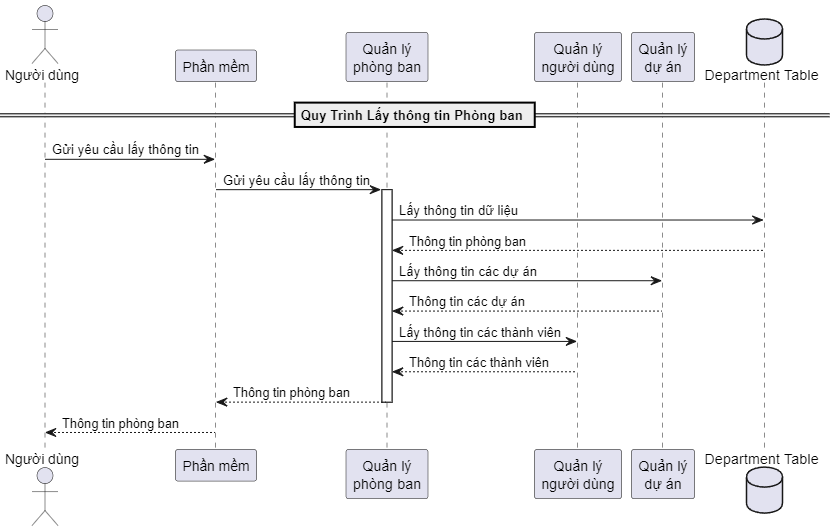
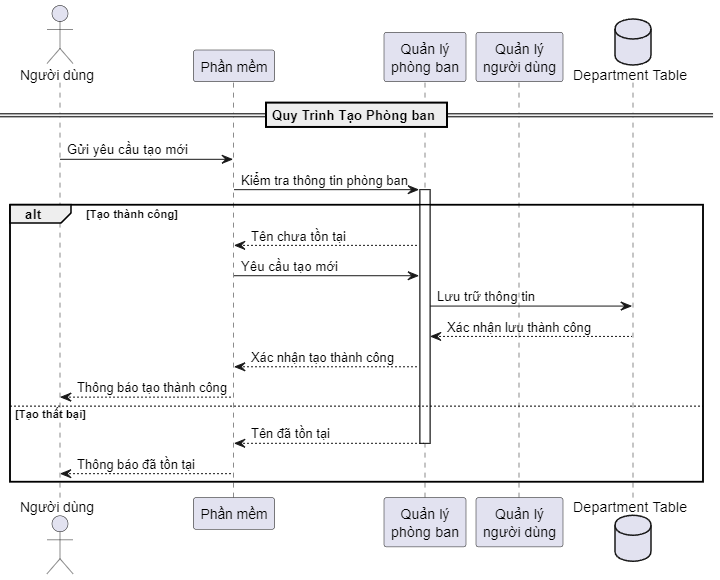
***3.2.2 Quản lý phòng ban:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp Department: chứa thông tin phòng ban và các phương thức để tương tác các lớp khác. Các phương thức:
  + create(departmentData: Department): tạo mới thông tin một phòng ban xuống database.
  + update(updateData: Department): cập nhật thông tin mới cho 1 phòng ban.
  + delete(id: Long): xóa 1 phòng ban theo phòng ban id.
  + getDepartmentInfo(): lấy thông tin 1 phòng ban.
  + listMembers(): thực hiện lấy danh sách các thành viên trong phòng ban.
  + listProjects(): thực hiện lấy danh sách các dự án được triển khai ở phòng ban.
* Lớp Project: thuộc phân hệ quản lý dự án, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa dự án và phòng ban.
* Lớp User: thuộc phân hệ quản lý người dùng, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa người dùng và phòng ban.



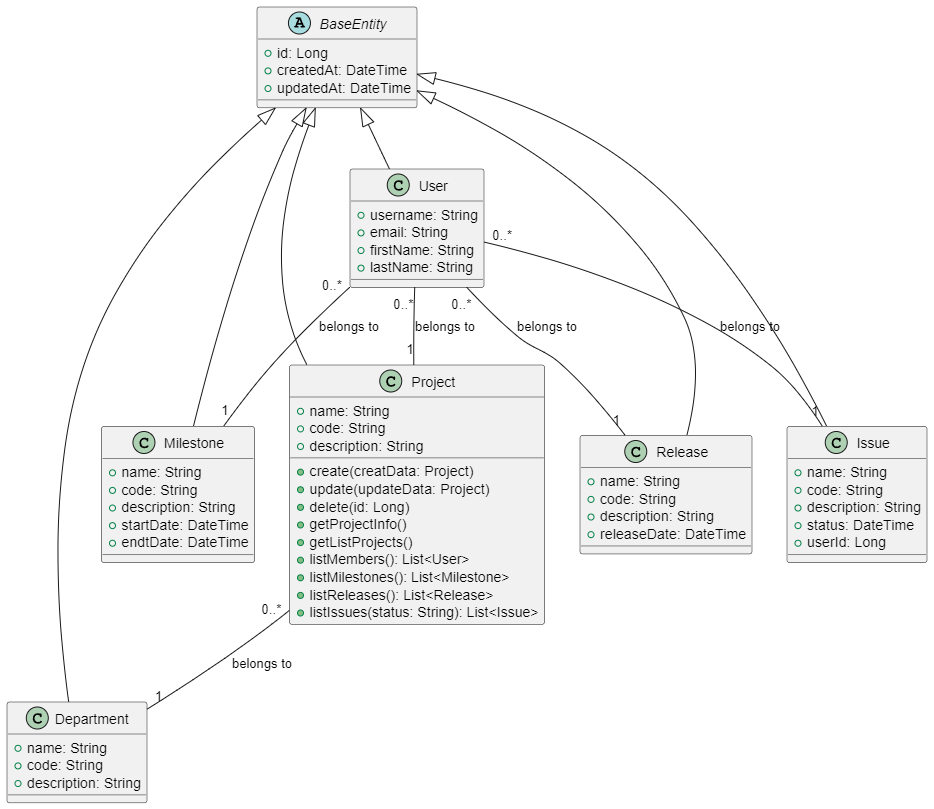
- Sơ đồ tuần tự:



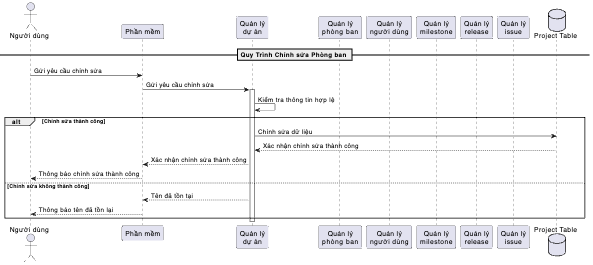
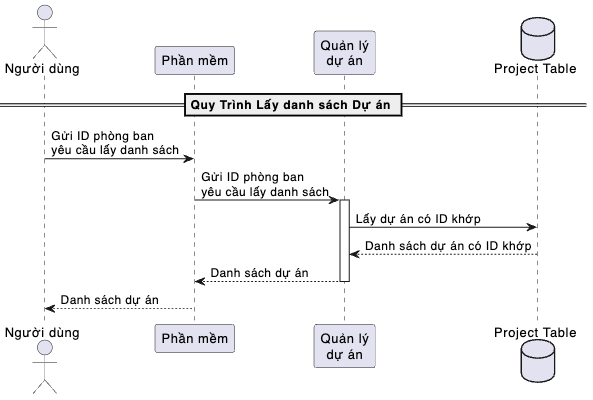
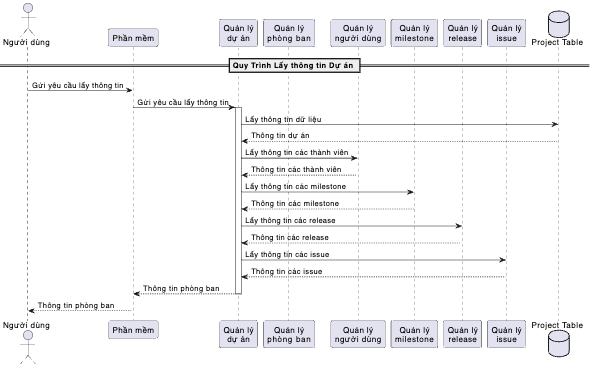
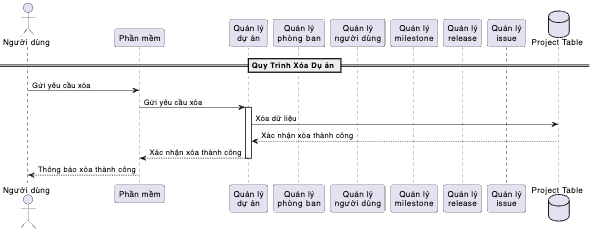
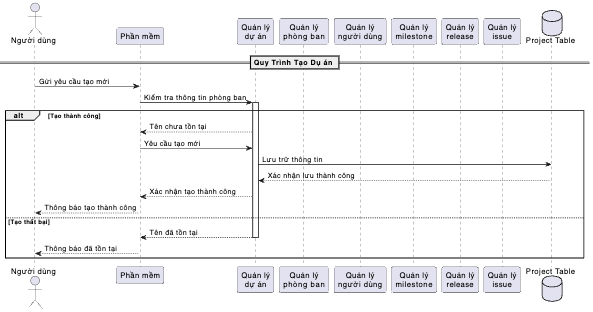
***3.2.3 Quản lý dự án:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp Project: chứa thông tin dự án và các phương thức để tương tác các lớp khác. Các phương thức:
  + create(creatData: Project): Tạo mới một dự án trong database.
  + update(updateData: Project): Cập nhật thông tin một dự án vào database.
  + delete(id: Long): Xóa một dự án từ id dự án.
  + getProjectInfo(): Lấy thông tin một dự án.
  + getListProjects(): Lấy danh sách các dự án.
  + listMembers(): List<User> Lấy danh sách các thành viên của dự án.
  + listMilestones(): List<Milestone> Lấy danh sách các milestones của dự án
  + listReleases(): List<Release> Lấy danh sách các releases của dự án.
  + listIssues(status: String): List<Issue> Lấy danh sách các issues của dự án.
* Lớp Department: thuộc phân hệ quản lý phòng ban, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa dự án và phòng ban
* Lớp User: thuộc phân hệ quản lý người dùng, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa người dùng và phòng ban.
* Lớp Milestone: thuộc phân hệ quản lý milestone, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa dự án và milestone.
* Lớp Release: thuộc phân hệ quản lý release, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa dự án và release.
* Lớp Issue: thuộc phân hệ quản lý issue, được đưa trong sơ đồ sau nhằm giải thích sự tương tác giữa dự án và issue.



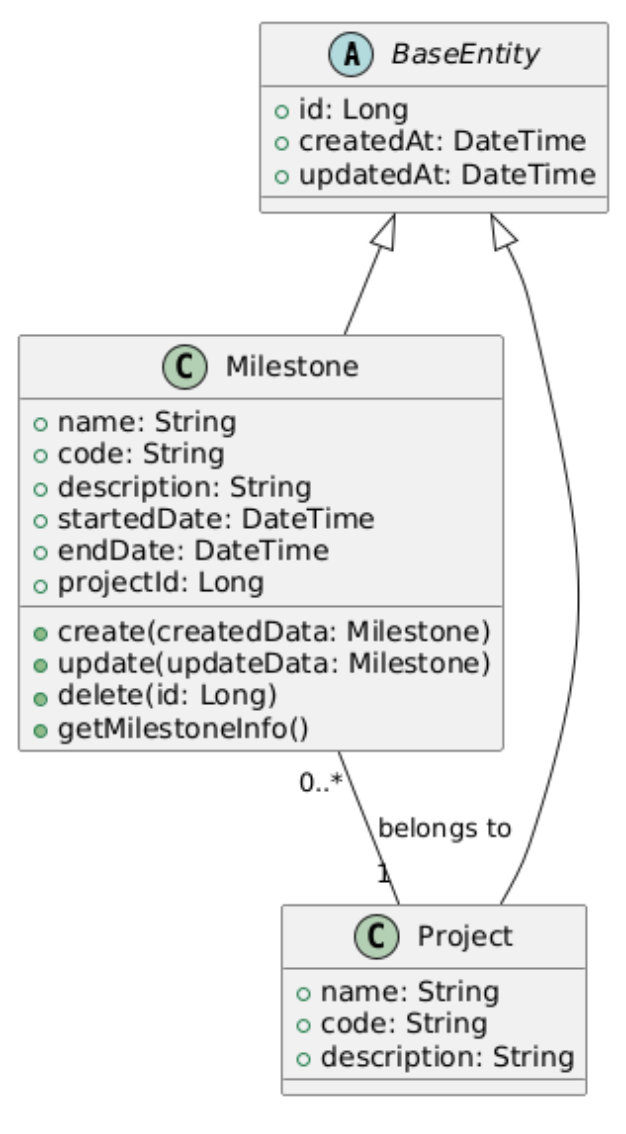
- Sơ đồ tuần tự:



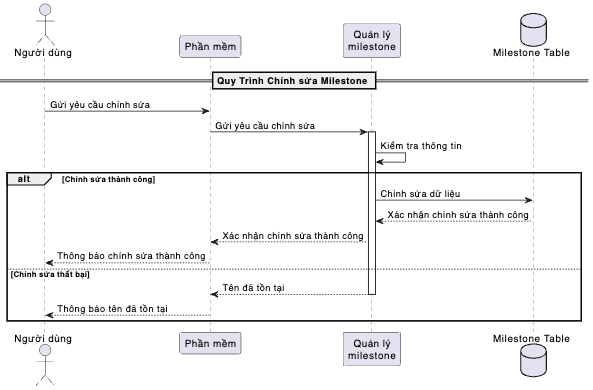
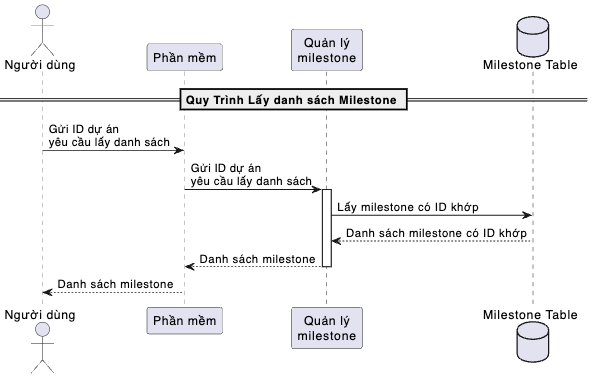
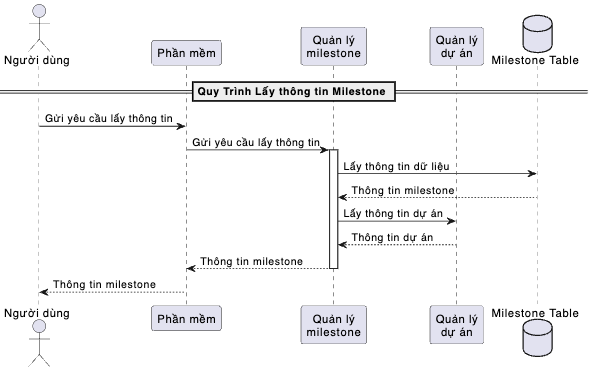
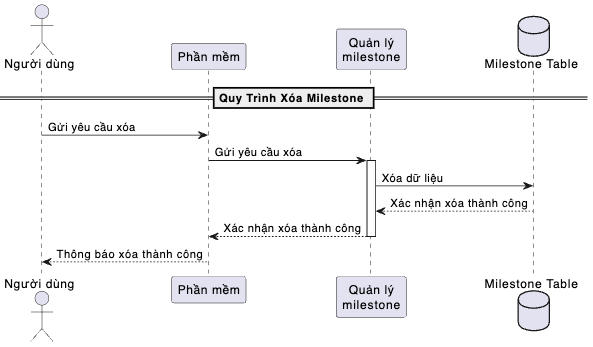
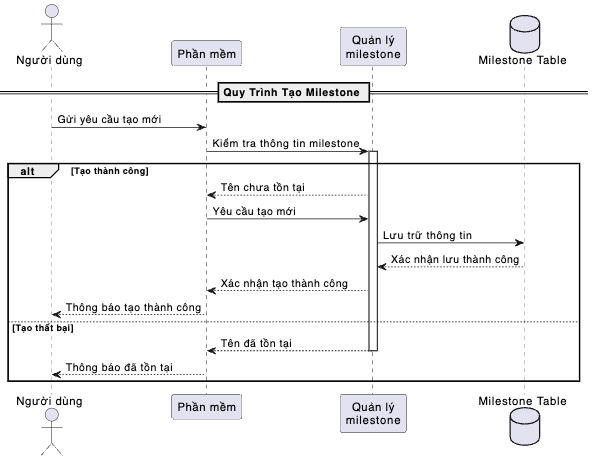
***3.2.4 Quản lý milestone:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp Project: thuộc phân hệ quản lý dự án, được đưa vào sơ đồ nhằm giải thích sự tương tác của milestone và dự án.
* Lớp Milestone: Lớp chứa các thuộc tính và phương thức để tương tác với các phân hệ khác. Các phương thức:
  + create(createdData: Milestone): Tạo mới một milestone vào database.
  + update(updateData: Milestone): Cập nhật lại thông tin milestone mới vào database.
  + delete(id: Long): Xóa một milestone với id của milestone.
  + getMilestoneInfo(): Lấy thông tin của một milestone.



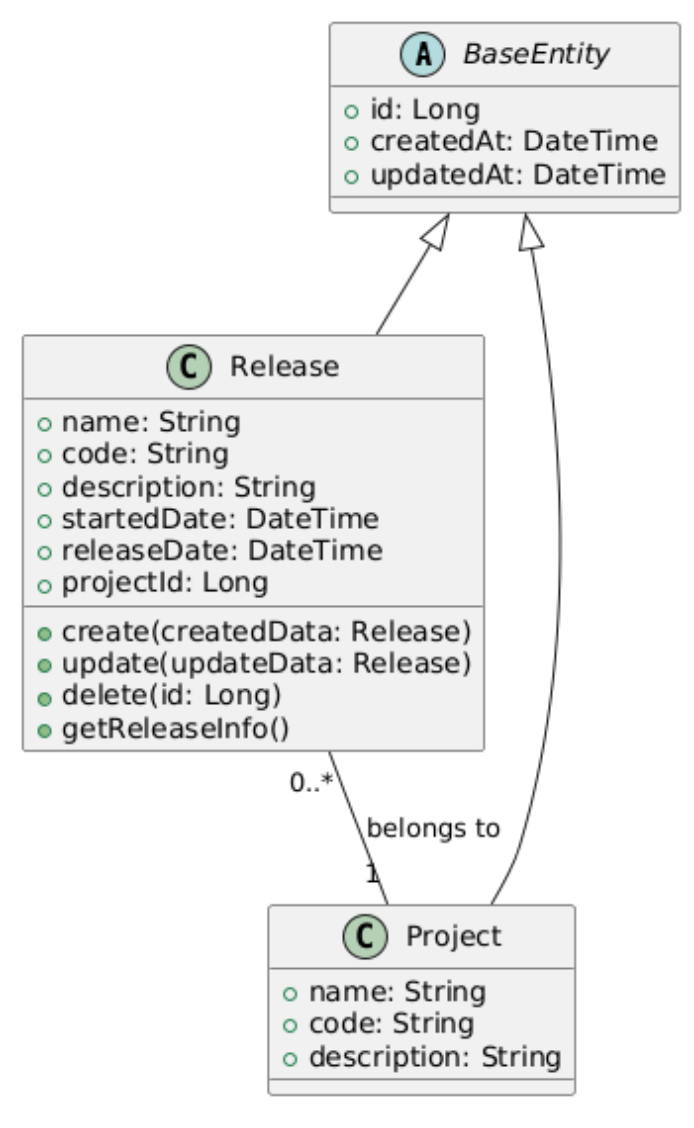
**-** Sơ đồ tuần tự:



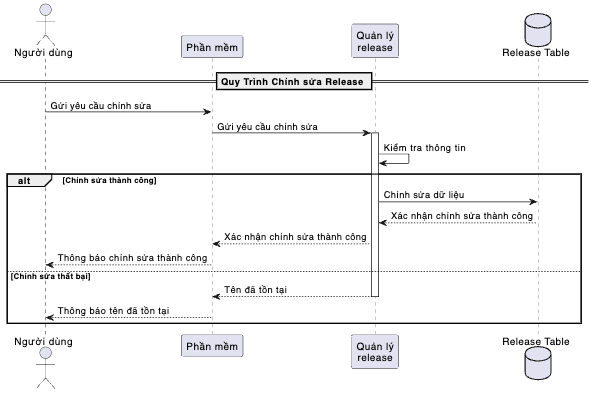
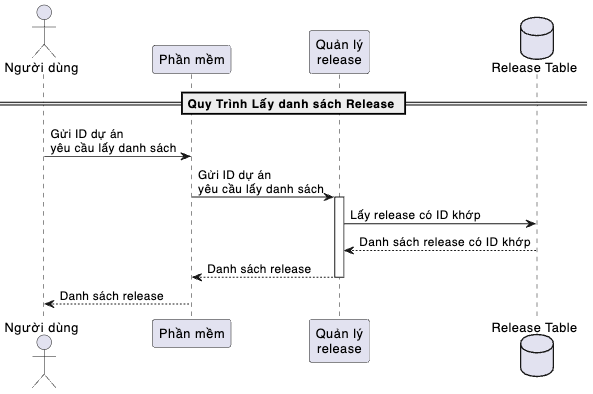
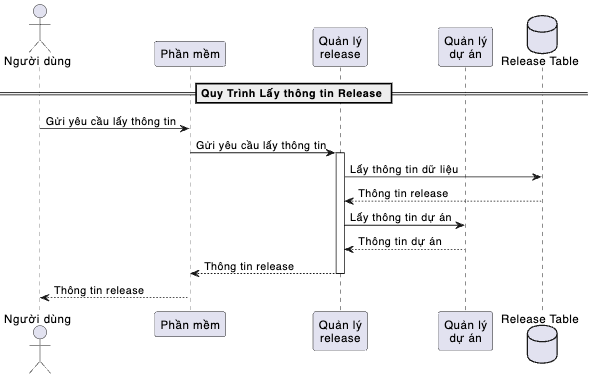
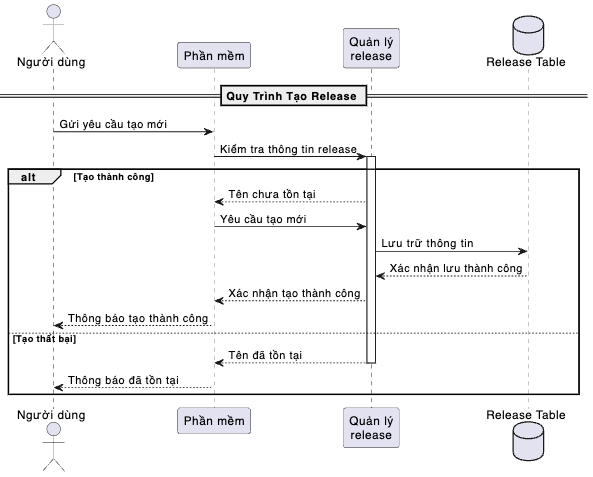
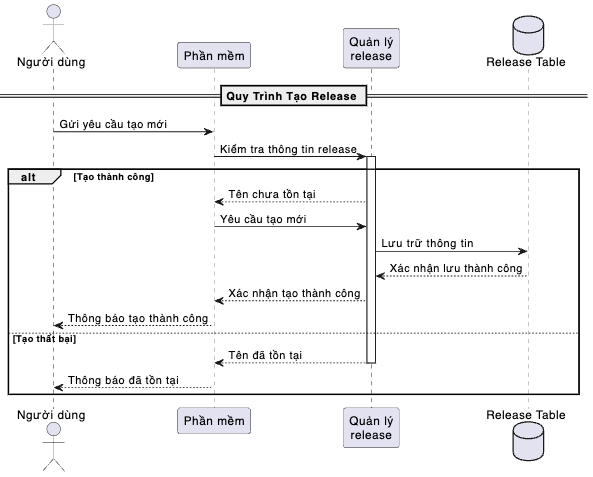
***3.2.5 Quản lý release:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp Project: thuộc phân hệ quản lý dự án, được đưa vào sơ đồ nhằm giải thích sự tương tác của release và dự án.
* Lớp Release: Lớp chứa các thuộc tính và phương thức để tương tác với các phân hệ khác. Các phương thức:
  + create(createdData: Release): Tạo mới một release vào database.
  + update(updateData: Release): Cập nhật lại thông tin release mới vào database.
  + delete(id: Long): Xóa một release với id của release.
  + getReleaseInfo(): Lấy thông tin của một release.



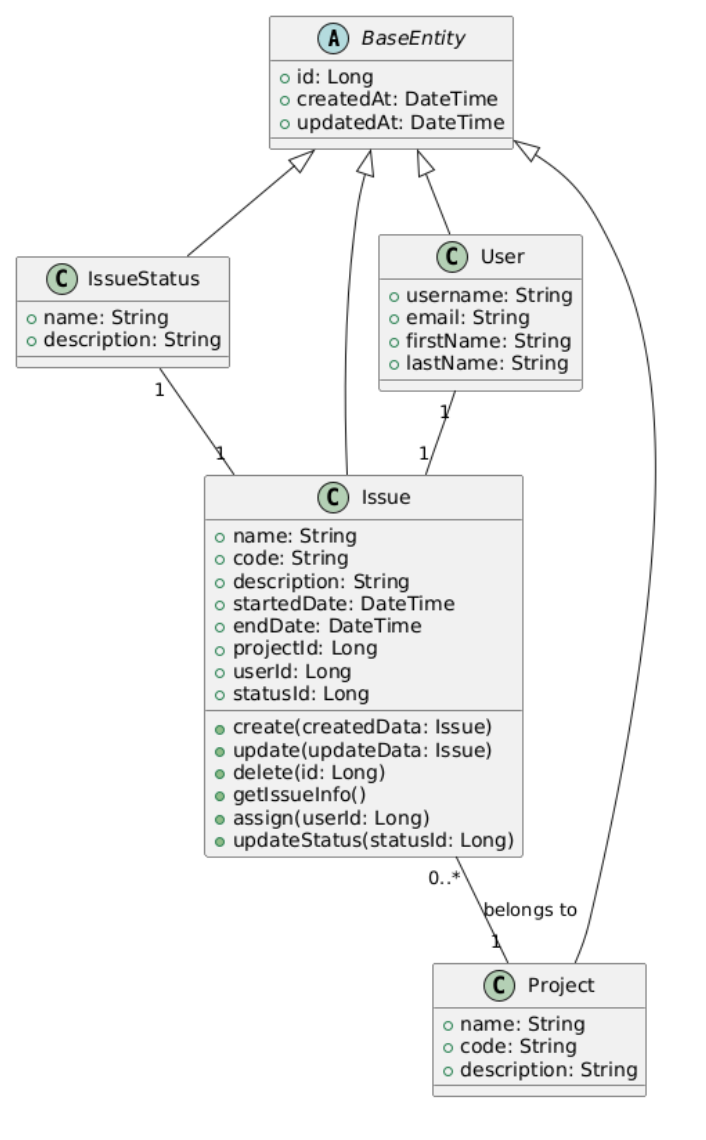
**-** Sơ đồ tuần tự:



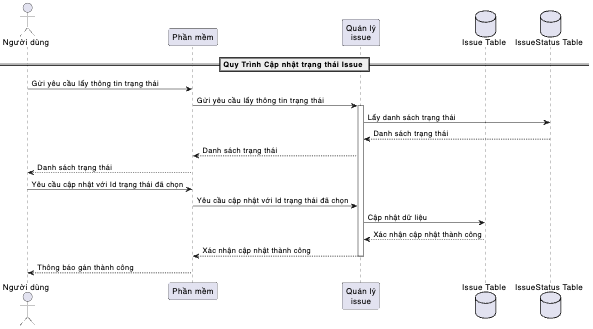
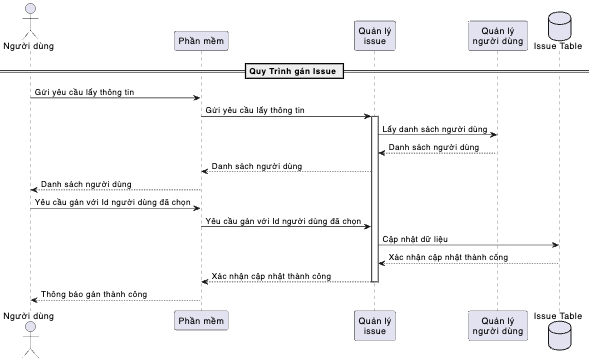
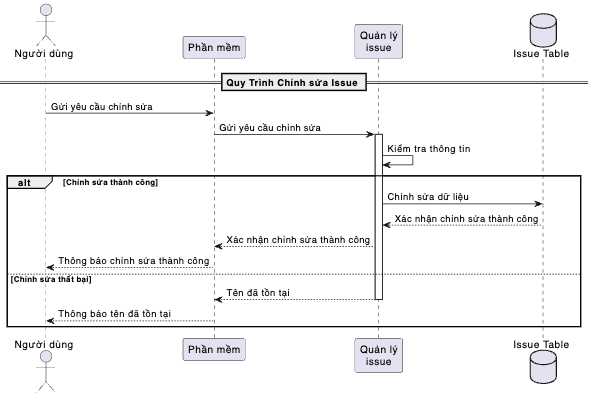
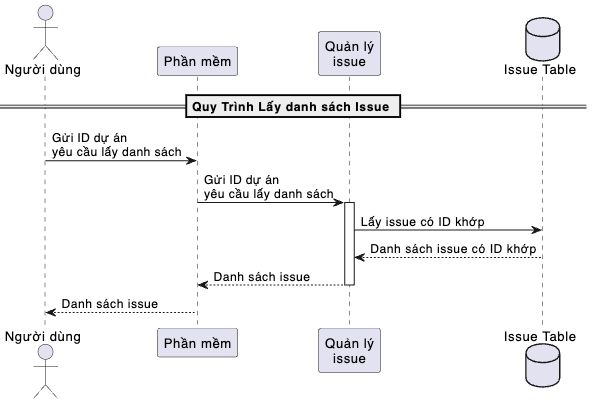
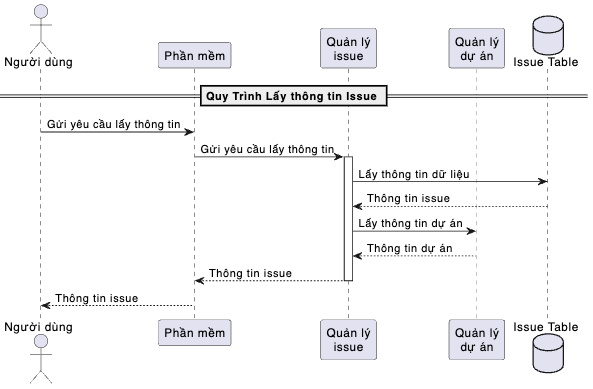
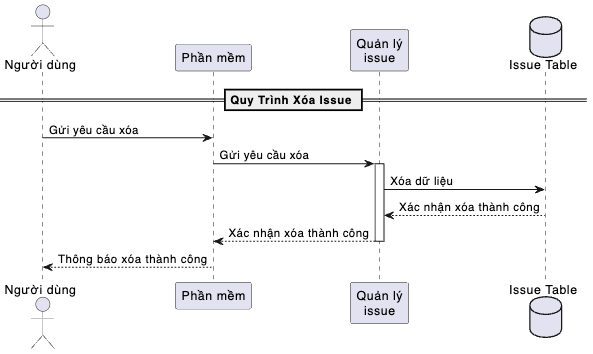
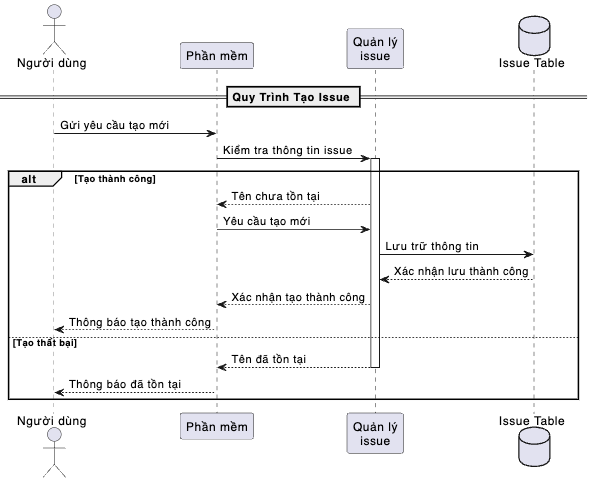
***3.2.6 Quản lý issue:***

**-** Các lớp trong thành phần:

* Lớp BaseEntity: chứa các thông tin cơ bản của một lớp để các lớp khác kế thừa.
* Lớp Project: thuộc phân hệ quản lý dự án, được đưa vào sơ đồ nhằm giải thích sự tương tác của issue và dự án.
* Lớp User: thuộc phân hệ quản lý người dùng, được đưa vào sơ đồ nhằm giải thích sự tương tác của issue và người dùng.
* Lớp Issue: Lớp chứa các thuộc tính và phương thức của issue để tương tác với các phân hệ khác. Các phương thức:
  + create(createdData: Issue): Tạo mới một issue vào database.
  + update(updateData: Issue): Cập nhật lại thông tin issue mới vào database.
  + delete(id: Long): Xóa một issue với id của issue.
  + getIssueInfo (): Lấy thông tin của một issue.
  + assign(userId: Long): Gán một issue cho một user bằng userId
  + updateStatus(statusId: Long): Cập nhật trạng thái issue bằng statusId.
* Lớp IssueStatus: Lớp bổ trợ chứa các trạng thái của issue.



**-** Sơ đồ tuần tự:



# 4. THIẾT KẾ DỮ LIỆU

## 4.1. Mô tả dữ liệu

1. Chuyển đổi miền thông tin sang cấu trúc dữ liệu:

- Quản lý người dùng:

* Thông tin nhân sự được chuyển đổi thành các đối tượng User.
* Thuộc phòng ban nào nào được thể hiện qua thuộc tính Id của đối tượng Department.
* Có vai trò gì trong phòng ban được thể hiện qua thuộc tính Id của đối tượng phụ Role.

- Quản lý phòng ban:

* Thông tin phòng ban được chuyển đổi thành các đối tượng Department.

- Quản lý dự án:

* Thông tin dự án được chuyển đổi thành các đối tượng Project.
* Được triển khai bởi phòng ban nào được thể hiện qua thuộc tính Id của đối tượng Department.

- Quản lý milestone:

* Thông tin milestone được chuyển đổi thành các đối tượng Milestone.
* Thuộc về dự án nào được thể hiện qua thuộc tính Id của đối tượng Project.

- Quản lý release:

* Thông tin release được chuyển đổi thành các đối tượng Release.
* Thuộc về dự án nào được thể hiện qua thuộc tính Id của đối tượng Project.

- Quản lý issue:

* Thông tin issue được chuyển đổi thành các đối tượng Issue.
* Trạng thái, phân công và quản lý bởi dự án được thể hiện qua các thuộc tính và đối tượng phụ là IssueStatus.

2. Cách thức lưu trữ dữ liệu:

- Cơ sở dữ liệu:

* Thiết kết theo mô hình dữ liệu quan hệ để đảm bảo tính toàn vẹn và hiệu quả.
* Lựa chọn hệ cơ sở dữ liệu PostgreSQL.

## 4.2. Từ điển dữ liệu

[Đưa ra theo thứ tự alphabet các thực thể dữ liệu hoặc các dữ liệu chính của phần mềm cùng với kiểu và sự mô tả của chúng. Cần đưa ra danh sách các đối tượng (lớp), thuộc tính, phương thức của các đối tượng (lớp) cùng với các tham số]

* Danh sách các đối tượng:
  + Department
  + Issue
  + IssueStatus
  + Milestone
  + Project
  + Release
  + Role
  + User
* Mô tả các đối tượng:
  + Department
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String : tên đối tượng
      * code: String : mã đối tượng dùng cho user đọc
      * description: String : mô tả đối tượng
    - Phương thức:
      * create(departmentData: Department): tạo mới một đối tượng
      * update(updateData: Department): cập nhật một đối tượng
      * delete(id: Long): xóa một đối tượng
      * getDepartmentInfo(): lấy thông tin một đối tượng
      * listMembers(): lấy danh sách thành viên của đối tượng
      * listProjects(): lấy danh sách dự án của đối tượng
  + Issue
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên của đối tượng
      * code: String: mã của đối tượng dành cho người dùng đọc
      * description: String: mô tả của đối tượng
      * startedDate: DateTime: ngày bắt đầu của đối tượng
      * endDate: DateTime: ngày kết thúc của đối tượng
      * projectId: Long: thể hiện dự án đối tượng đang thuộc về
      * userId: Long: thể hiện đối tượng đang được gán cho user nào
      * statusId: Long: thể hiện trạng thái của đối tượng
    - Phương thức:
      * create(createdData: Issue): tạo một đối tượng
      * update(updateData: Issue): cập nhật đối tượng
      * delete(id: Long): xóa đối tượng
      * getMilestoneInfo(): lấy thông tin của đối tượng
      * assign(userId: Long): gán đối tượng cho một user
      * updateStatus(statusId: Long): cập nhật trạng thái của đối tượng
  + IssueStatus
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên của đối tượng
      * description: String: mô tả đối tượng
  + Milestone
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên của đối tượng
      * code: String: mã của đối tượng dành cho người dùng đọc
      * description: String: mô tả đối tượng
      * startedDate: DateTime: ngày bắt đầu đối tượng
      * endDate: DateTime: ngày kết thúc đối tượng
      * projectId: Long: dự án đối tượng thuộc về
    - Phương thức:
      * create(createdData: Milestone): tạo mới đối tượng
      * update(updateData: Milestone): cập nhật đối tượng
      * delete(id: Long): xóa đối tượng
      * getMilestoneInfo(): lấy thông tin đối tượng
  + Project
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên đối tượng
      * code: String: mã đối tượng dành cho người dùng đọc
      * description: String: mô tả đối tượng
    - Phương thức:
      * create(creatData: Project): tạo đối tượng
      * update(updateData: Project): cập nhật đối tượng
      * delete(id: Long): xóa đối tượng
      * getProjectInfo(): lấy thông tin đối tượng
      * getListProjects(): lấy danh sách các đối tượng
      * listMembers(): lấy danh sách các thành viên của đối tượng
      * listMilestones(): lấy danh sách các milestones của đối tượng
      * listReleases(): lấy danh sách các releases của đối tượng
      * listIssues(status: String): lấy danh sách các issues của đối tượng
  + Release
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên của đối tượng
      * code: String: mã của đối tượng dành cho người dùng đọc
      * description: String: mô tả đối tượng
      * startedDate: DateTime: ngày bắt đầu đối tượng
      * releaseDate: DateTime: ngày kết thúc đối tượng
      * projectId: Long: dự án đối tượng thuộc về
    - Phương thức:
      * create(createdData: Release): tạo mới đối tượng
      * update(updateData: Release): cập nhật đối tượng
      * delete(id: Long): xóa đối tượng
      * getReleaseInfo(): lấy thông tin đối tượng
  + Role
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * name: String: tên đối tượng
      * description: String: mô tả đối tượng
  + User
    - Thuộc tính:
      * id: Long: thể hiện mã đơn nhất cho mỗi đối tượng
      * createdAt: DateTime: ngày tháng tạo đối tượng
      * updatedAt: DateTime: ngày tháng cập nhật đối tượng
      * username: String: tên đăng nhập của đối tượng
      * email: String: email của đối tượng
      * passwordHash: String: password ở dạng hash của đối tượng
      * firstName: String: tên của đối tượng
      * lastName: String: tên lót của đối tượng
      * phoneNumber: String: số điện thoại của đối tượng
      * isActive: Boolean: trạng thái hoạt động của đối tượng
      * roleId: Long: phân quyền của đối tượng
      * departmentId: Long: phòng ban đối tượng thuộc về
    - Phương thức:
      * checkUserNameExist(userName: String): kiểm tra xem tên đăng nhập của đối tượng đã tồn tại chưa
      * create(userInfo: User): tạo mới đối tượng
      * hashPassword(password: String): hash password của đối tượng
      * changePassword(newPassword: String): cập nhật password của đối tượng
      * getProfile(id: Long): lấy thông tin của đối tượng
      * updateProfile(firstName:String,lastName:String, phoneNumber: String): cập nhật thông tin đối tượng
      * activate(): cập nhật trạng thái hoạt động cho đối tượng
      * deactivate(): cập nhật trạng thái không hoạt động cho đối tượng
      * updateRole(roleId: Long): cập nhật phân quyền cho đối tượng
      * updateDepartment(departmentId: Long): cập nhật phòng ban đối tượng thuộc về
      * login(username: String, password: String): đăng nhập đối tượng vào hệ thống