**Trường Đại Học Công Nghiệp TP.HCM**

**Khoa Công Nghệ Thông Tin**

**------**

Icon

Description automatically generated with medium confidence

**Đề tài**: Thanh lý tái chế điện thoại

**Nhóm 8**:

-Đỗ Nguyên Khang 20059731

-Lê Ngọc Hoàng 20056961

-Nguyễn Hoàng Anh 20060251

**Báo cáo tiểu luận**

**Môn: Kiến trúc và thiết kế phần mềm**

**Gv: Võ Văn Hải**

# 1. Tiêu đề (Title)

Một tiêu đề rõ ràng và hấp dẫn, phản ánh nội dung chính của bài tiểu luận.

- Thanh lý điện thoại.

# 2. Mục lục (Table of Contents)

Liệt kê các phần chính và các mục nhỏ của bài tiểu luận cùng với số trang tương ứng.

# 3. Giới thiệu (Introduction)

## Tại sao cần thiết kế phần mề thanh lí tái chế điện thoại.

Công nghệ ngày càng phát triển những sản phẩm mới ngày càng tiên tiến và cập nhật không ngừng nghỉ cho nên để cho việc muốn sử dụng các sản phẩm mới mà không cần bỏ đi các sản phẩm cũ của mình cho nên nhóm em đã làm ra phần mềm thanh lí tái chế điện thoại để giúp cho các sản phẩm không bị bỏ đi một cách đáng tiếc ngoài ra còn:

* **Bảo vệ môi trường**
* Giảm thiểu rác thải : Điện thoại di động và các thiết bị điện tử khác chứa nhiều thành phần có thể gây hại cho môi trường nếu không được xử lý đúng cách. Việc tái chế giúp giảm lượng rác thải điện tử, ngăn chặn ô nhiễm đất và nước.
* Tái sử dụng linh kiện: Các linh kiện điện tử còn hoạt động tốt có thể được thu hồi và tái sử dụng, giảm nhu cầu khai thác tài nguyên mới và giảm thiểu tác động tiêu cực đến môi trường.
* **Tiết kiệm tài nguyên**
* Khôi phục giá trị từ các thiết bị cũ: Nhiều linh kiện và vật liệu trong điện thoại cũ vẫn còn giá trị. Tái chế giúp khai thác tối đa các tài nguyên này, từ kim loại quý như vàng, bạc đến các thành phần như nhựa và thủy tinh.
* Giảm chi phí sản xuất: Tái sử dụng linh kiện có thể giúp giảm chi phí sản xuất thiết bị mới, tạo lợi thế cạnh tranh cho các nhà sản xuất.
* **Lợi ích kinh tế**
* Tạo nguồn thu nhập: các thiết bị cũ có thể bán đi, tạo ra nguồn thu nhập cho cá nhân và doanh nghiệp.
* Tạo việc làm: Ngành tái chế điện tử cần một lượng lớn nhân lực cho các công việc như phân loại, sửa chữa, và tái chế, tạo ra cơ hội việc làm cho nhiều người.
* **Nâng cao trải nghiệm của khách hàng**
* Dịch vụ khách hàng tốt hơn: Khách hàng có thể dễ dàng gửi thiết bị cũ để tái chế, theo dõi quá trình xử lý, và nhận thông tin về việc mua lại các sản phẩm đã qua tái chế với giá hợp lý.
* Khuyến khích tái chế: Tạo ra các chương trình khuyến khích khách hàng tham gia vào việc tái chế, từ đó nâng cao ý thức bảo vệ môi trường trong cộng đồng.

## **Mục tiêu: Mục tiêu chính của bài tiểu luận này là gì.**

* **Tạo Nền Tảng Giao Dịch Đáng Tin Cậy:**

- Xây dựng một nền tảng mà người dùng có thể yên tâm mua bán điện thoại cũ với sự đảm bảo về chất lượng và tính chính xác của thông tin.

* **Đơn Giản Hóa Quy Trình Mua Bán:**

- Cung cấp giao diện người dùng thân thiện và quy trình đơn giản để người bán có thể dễ dàng đăng tin và người mua có thể dễ dàng tìm kiếm, so sánh và mua sản phẩm.

* **Tăng Tính Minh Bạch:**

- Đảm bảo rằng tất cả các thông tin về điện thoại được đăng bán đều rõ ràng và minh bạch, bao gồm hình ảnh thật, tình trạng máy, và lịch sử sử dụng.

* **Đảm Bảo An Toàn Giao Dịch:**

- Tích hợp các phương thức thanh toán an toàn và cung cấp các biện pháp bảo vệ người mua và người bán để giảm thiểu rủi ro trong quá trình giao dịch.

* **Xây Dựng Cộng Đồng Người Dùng:**

- Khuyến khích người dùng tham gia đánh giá, phản hồi và chia sẻ kinh nghiệm để tạo ra một cộng đồng tích cực, hỗ trợ lẫn nhau trong việc mua bán điện thoại cũ.

* **Tối Ưu Hóa Tính Năng Tìm Kiếm và Lọc:**

- Cấp các công cụ tìm kiếm và bộ lọc mạnh mẽ giúp người dùng dễ dàng tìm thấy sản phẩm phù hợp với nhu cầu và ngân sách của mình.

* **Hỗ Trợ Người Dùng:**

- Cung cấp dịch vụ hỗ trợ khách hàng nhanh chóng và hiệu quả, giúp giải quyết các vấn đề phát sinh trong quá trình sử dụng ứng dụng.

* **Khuyến Khích Bảo Vệ Môi Trường:**

- Thúc đẩy việc tái sử dụng điện thoại cũ, giảm thiểu rác thải điện tử, và nâng cao nhận thức về bảo vệ môi trường thông qua việc sử dụng và thanh lý thiết bị cũ.

* **Tích Hợp Các Tính Năng Đa Dạng:**

- Cung cấp các tính năng phụ trợ như kiểm tra giá thị trường, thông báo khi có sản phẩm mới phù hợp, và so sánh giá cả để người dùng có thể đưa ra quyết định mua bán tốt nhất.

* **Phát Triển Bền Vững:**

- Đảm bảo mô hình kinh doanh bền vững để ứng dụng có thể hoạt động lâu dài, cung cấp giá trị liên tục cho người dùng và duy trì sự phát triển của nền tảng.

## Phạm vi hoạt động:

Sử dụng trên các nền tảng web .

## 4.Phạm vi đề tài:

**Chức năng cho người dùng**

* Giao diện dễ sử dụng và thân thiện, dễ thao tác và sử dụng.
* Cho phép người dùng đăng kí tài khoản và quên mật khẩu.
* Cho phép người dùng thanh lý sản phẩm của mình.

**Chức năng dành cho admin.**

Ngoài các yêu cầu giống như của người dùng, thì hệ thống phải đảm bảo những yêu cầu sau:

* Quản lý tài khoản của người dùng .
* Phê duyệt sản phẩm của người dùng.

# 4. Kiến trúc phần mềm (Software Architecture)

## Định nghĩa và giới thiệu

**Kiến trúc phần mềm là gì ?**

- Kiến trúc phần mềm: là một khía cạnh quan trọng trong lĩnh vực phát triển phần mềm, đại diện cho cấu trúc tổng thể của một hệ thống phần mềm. Nó bao gồm các thành phần phần mềm, cách các thành phần này tương tác với nhau, và các nguyên tắc, chuẩn mực được sử dụng trong thiết kế và triển khai hệ thống.

* + - * 1. **Thành phần phần mềm (Software Components):**

- **Modules (Mô-đun):** Các khối xây dựng cơ bản của phần mềm, mỗi mô-đun thực hiện một chức năng cụ thể.

- **Layers (Lớp):** Các lớp chức năng, ví dụ như Presentation Layer (lớp trình diễn), Business Logic Layer (lớp logic nghiệp vụ), và Data Access Layer (lớp truy cập dữ liệu).

- **Services (Dịch vụ):** Các thành phần độc lập thực hiện các chức năng nhất định và giao tiếp qua các giao diện xác định.

* + - * 1. **Kết nối (Connectors):**

- Các phương thức hoặc giao diện mà qua đó các thành phần phần mềm giao tiếp và tương tác với nhau. Ví dụ: API, giao thức HTTP, hệ thống nhắn tin.

* + - * 1. **Phong cách kiến trúc (Architectural Styles):**

- Các mô hình hoặc kiểu thiết kế mà kiến trúc phần mềm có thể tuân theo, chẳng hạn như kiến trúc phân lớp (Layered Architecture), kiến trúc microservices, kiến trúc hướng dịch vụ (SOA), và kiến trúc dựa trên sự kiện (event-driven architecture).

* + - * 1. **Các quyết định thiết kế (Design Decisions):**

- Các quyết định quan trọng ảnh hưởng đến cấu trúc tổng thể của hệ thống, bao gồm việc chọn lựa công nghệ, các mẫu thiết kế (design patterns), và các chiến lược triển khai.

* + - * 1. **Chất lượng dịch vụ (Quality Attributes):**

- Các thuộc tính phi chức năng của hệ thống như hiệu năng, bảo mật, khả năng mở rộng, khả năng bảo trì, và độ tin cậy.

## Một số loại kiến trúc phần mềm phổ biến

**(a)Monolithic Architecture (Kiến trúc nguyên khối)**

Tất cả các thành phần của hệ thống được đóng gói vào một ứng dụng duy nhất.

Ưu điểm: Dễ triển khai, dễ quản lý ban đầu.

Nhược điểm: Khó mở rộng, khó bảo trì và dễ gặp vấn đề về độ tin cậy.

**(b)Microservices Architecture (Kiến trúc vi dịch vụ)**

Hệ thống được chia thành nhiều dịch vụ nhỏ, mỗi dịch vụ chịu trách nhiệm một phần chức năng riêng biệt.

Ưu điểm: Dễ mở rộng, dễ bảo trì, độ tin cậy cao.

Nhược điểm: Phức tạp hơn trong quản lý và triển khai, yêu cầu quản lý giao tiếp giữa các dịch vụ.

**(c)Service-Oriented Architecture (SOA - Kiến trúc hướng dịch vụ)**

Các dịch vụ được triển khai độc lập và giao tiếp thông qua giao diện được xác định trước.

Tương tự như kiến trúc vi dịch vụ nhưng thường được áp dụng ở mức độ doanh nghiệp lớn hơn.

Ưu điểm: Tái sử dụng dịch vụ dễ dàng, linh hoạt.

Nhược điểm: Đòi hỏi quy hoạch và quản lý tốt, có thể phức tạp trong triển khai.

**(d)Layered (Tiered) Architecture (Kiến trúc phân lớp)**

Hệ thống được chia thành các lớp với chức năng riêng biệt (ví dụ: Presentation, Business Logic, Data Access).

Ưu điểm: Dễ phát triển và bảo trì, phân tách rõ ràng các mối quan tâm.

Nhược điểm: Có thể trở nên cồng kềnh và khó mở rộng khi quy mô lớn.

**(e)Pipeline Architecture(Kiến trúc đường ống)**

Ưu điểm: Dễ mở rộng và phát triển, tái sử dụng lại các pipeline khác, phân tách công việc.

Nhược điểm: chỉ cần một pipeline bị chậm sẽ ảnh hưởng đến toan bộ pipeline khác, phức tạp hóa trong quá trình đồng bộ hóa.

**(f) Mô hình Model-View-Controller (MVC)**

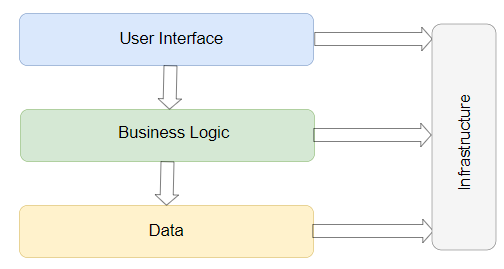
Ưu điểm: bảo trì code dễ dàng , dễ dàng mở rộng và phát triển.việc phát triển các thành phần khác nhau có thể thực hiện song song.

Nhược điểm:Không có hỗ trợ xác thực chính thức , bảo trì nhiều code trong Controller.

## Phần mềm được sử dụng loại kiến trúc nào ?

Được thiết kế theo loại kiến trúc Layered (Tiered) Architecture (Kiến trúc phân lớp).

**Mô hình Layered Architecture**



-User Interface (UI): (Giao diện người dùng) là thành phần mà người dùng tương tác trực tiếp khi sử dụng một sản phẩm hoặc dịch vụ kỹ thuật số, như một trang web, ứng dụng di động, hoặc phần mềm. Mục tiêu của UI là tạo ra một trải nghiệm trực quan và hiệu quả, giúp người dùng dễ dàng thực hiện các tác vụ cần thiết

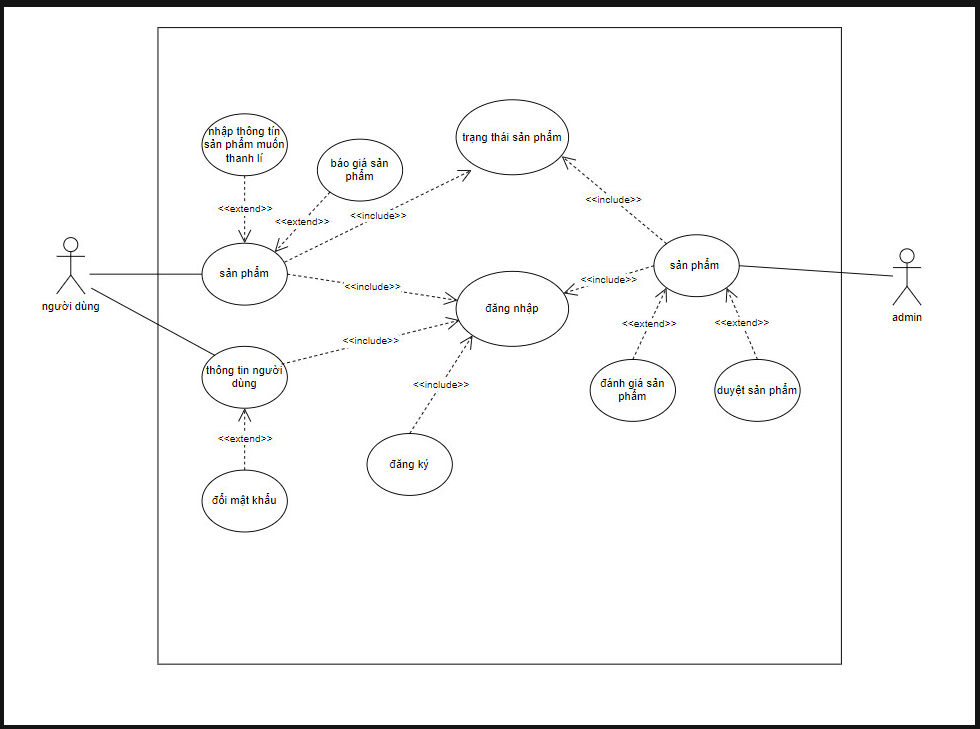
-Business Logic (Logic kinh doanh) :là một phần quan trọng trong thiết kế và phát triển phần mềm, chịu trách nhiệm xử lý các quy tắc và thao tác dữ liệu cụ thể của một ứng dụng liên quan đến nghiệp vụ của doanh nghiệp. Đây là lớp trung gian giữa giao diện người dùng (UI) và tầng dữ liệu (Database).

-Database (Cơ sở dữ liệu): là một tập hợp có tổ chức của dữ liệu, được lưu trữ và quản lý theo cách có thể truy cập, quản lý, và cập nhật dễ dàng. Cơ sở dữ liệu là nền tảng của nhiều hệ thống thông tin hiện đại, hỗ trợ lưu trữ và truy xuất thông tin một cách hiệu quả và đáng tin cậy.

-Infrastructure (Cơ sở hạ tầng) trong công nghệ thông tin (IT) đề cập đến tập hợp các thành phần phần cứng, phần mềm, mạng, và các dịch vụ cần thiết để hỗ trợ hoạt động, quản lý và phát triển của các hệ thống IT. Đây là nền tảng mà các ứng dụng, dịch vụ và hoạt động kinh doanh kỹ thuật số được xây dựng và vận hành.

# 5. Thiết kế phần mềm (Software Design)

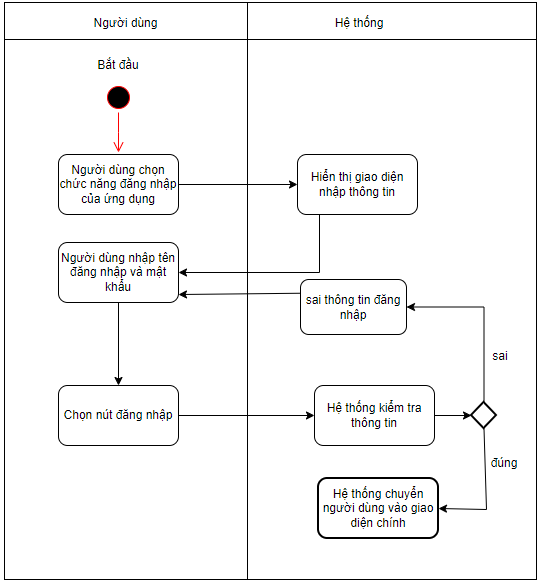
## 5.1. Sơ đồ use case tổng quát



## 5.2.Tình huống hoạt động

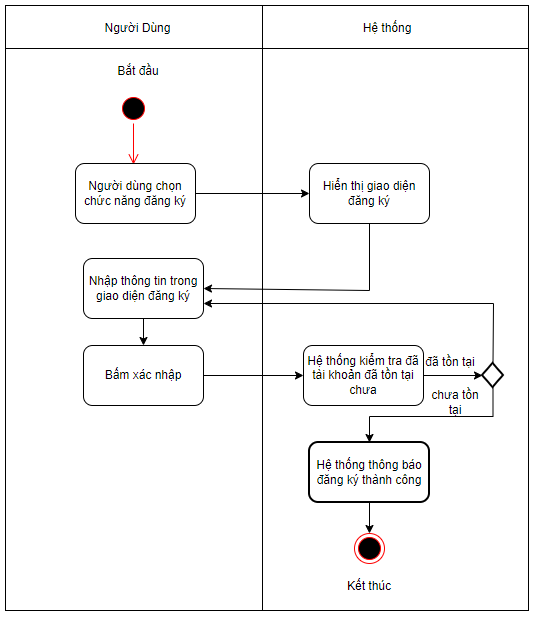
**5.2.1.Đăng nhập**

|  |  |
| --- | --- |
| * **Tên use case**: Đăng nhập | |
| * **Mô tả sơ lược**: Người dùng muốn đăng nhập vào hệ thống để sử dụng các dịch vụ | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Người dùng chọn chức năng đăng nhập trên trang chủ | 1. Hệ thống hiển thị giao diện nhập thông tin tài khoản |
| 1. Người dùng nhập tài khoản gồm sđt/email và mật khẩu |  |
| 1. Chọn nút đăng nhập | 1. Hệ thống kiểm tra thông tin nhập vào |
|  | 1. Hệ thống chuyển hướng người dùng vào giao diện chính của ứng dụng. |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 6.1. Hệ thống thông báo thông tin tài khoản đăng nhập không đúng |
|  | 6.2. Hệ thống quay lại bước 3 |



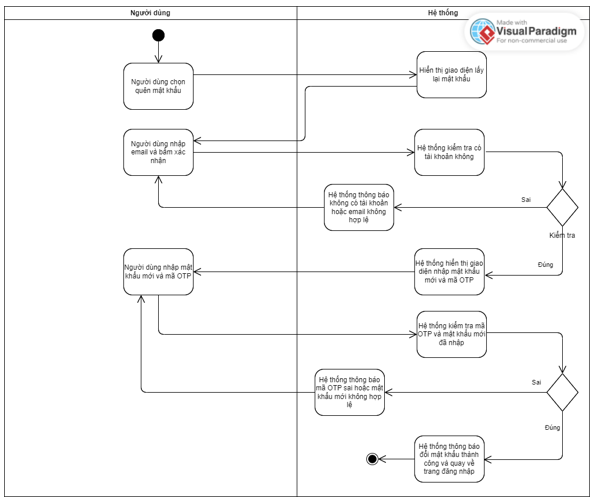
**5.2.2.Đăng ký**

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

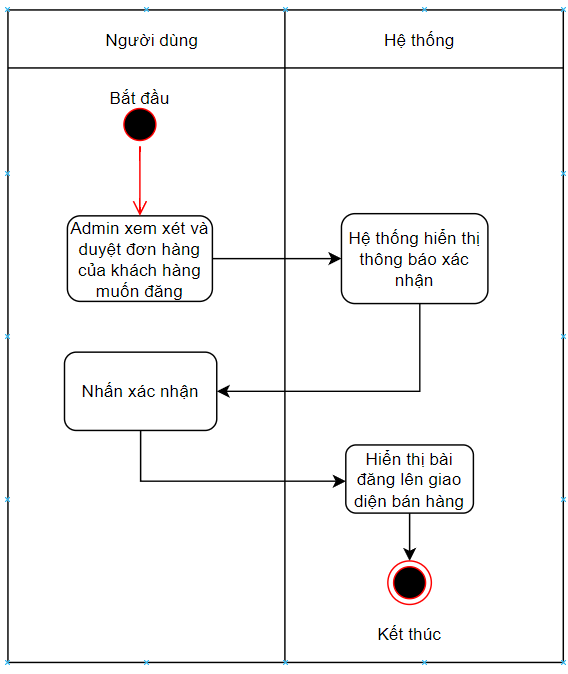


**5.2.3.Quên mật khẩu**

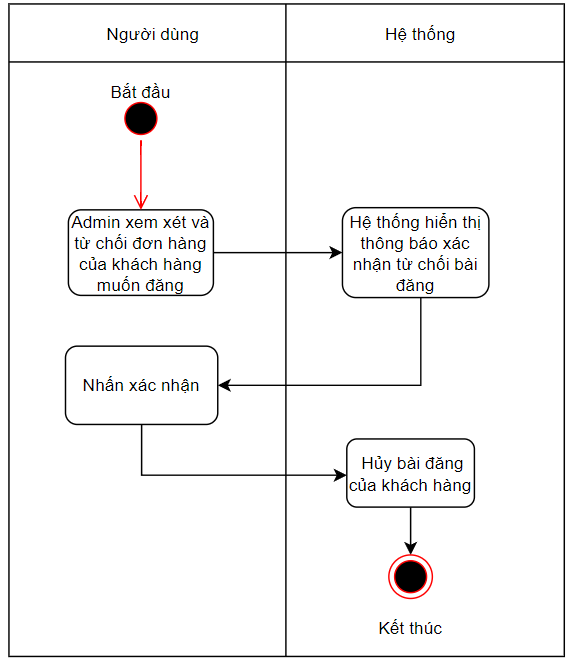
|  |  |
| --- | --- |
| * **Tên use case**: Quên mật khẩu | |
| * **Mô tả sơ lược**: Người dùng lấy lại mật khẩu đã mất | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** Người dùng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



|  |  |
| --- | --- |
| * **Tên use case:** Xét duyệt | |
| * **Mô tả sơ lược:** Kiểm tra và xét duyệt các đơn hàng của người dùng | |
| * **Actor chính:** Admin | |
| * **Actor phụ: Không** | |
| * **Tiền điều kiện (Pre-condition):** Admin phải đăng nhập vào hệ thống | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Admin xem xét đơn hàng của người dùng rồi bấm xét duyệt | 1. Hệ thống hiển thị giao diện xác nhận xét duyệt |
| 1. Admin bấm xác nhận | 1. Hệ thống thông báo xét duyệt thành công |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  |  |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |

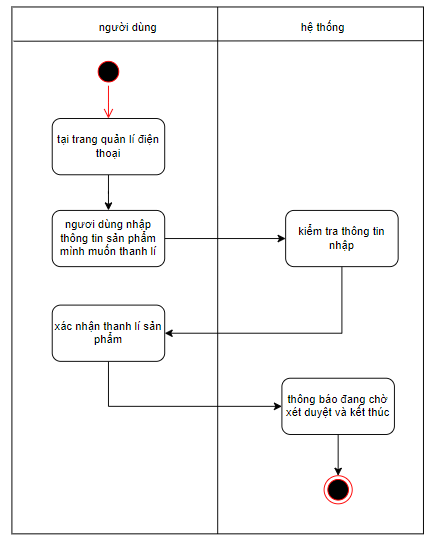


|  |  |
| --- | --- |
| * **Tên use case:** Hủy xét duyệt | |
| * **Mô tả sơ lược:** Kiểm tra và xét duyệt các đơn hàng của người dùng | |
| * **Actor chính:** Admin | |
| * **Actor phụ: Không** | |
| * **Tiền điều kiện (Pre-condition):** Admin phải đăng nhập vào hệ thống | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Admin xem xét đơn hàng của người dùng rồi bấm và từ chối xét duyệt. | 1. Hệ thống hiển thị giao diện xác nhận hủy bài đăng của khách hàng. |
| 1. Admin bấm xác nhận hủy. | 1. Hệ thống thông báo xét duyệt từ chối thành công. |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  |  |
| * **Luồng sự kiện ngoại lệ (exception flow):** | |



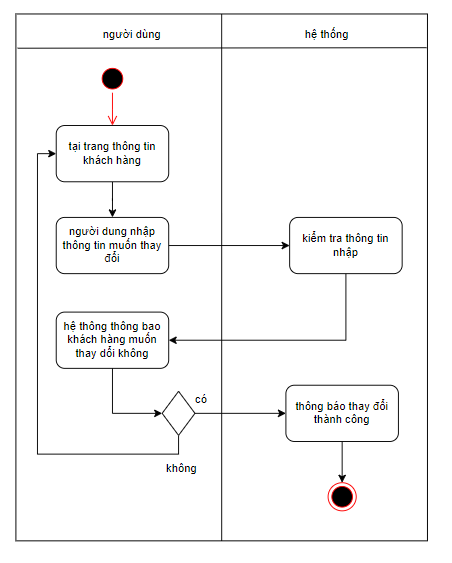
**5.2.4.Thanh lý**

|  |  |
| --- | --- |
| * **Tên use case**: UC01\_Thanh lý | |
| * **Mô tả sơ lược**: Người dùng muốn thanh lý sản phẩm của mình | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản và sản phẩm muốn bán | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. tại trang quản lí điện thoại |  |
| 1. ngươi dùng nhập thông tin sản phẩm mình muốn thanh lí | 1. kiểm tra thông tin nhập |
| 1. xác nhận thanh lí sản phẩm | 1. thông báo đang chờ xét duyệt và kết thúc |
|  |  |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  |  |



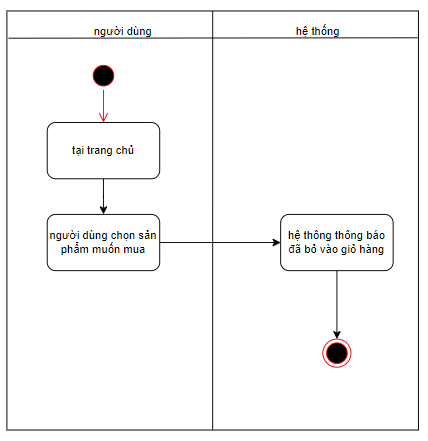
**5.2.5.Thay đổi thông tin**

|  |  |
| --- | --- |
| * **Tên use case**: UC01\_Thay đổi thông tin | |
| * **Mô tả sơ lược**: Người dùng thây đổi thông tin của mình | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. tại trang thông tin khách hàng |  |
| 1. người dung nhập thông tin muốn thay đổi | 1. kiểm tra thông tin nhập |
| 1. hệ thông thông bao khách hàng muốn thay dổi không | 1. cập nhât thành công và kết thúc |
|  |  |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4.1 thông báo cập nhật thành công |
|  | 4.2 quay trở lai bước 1 |



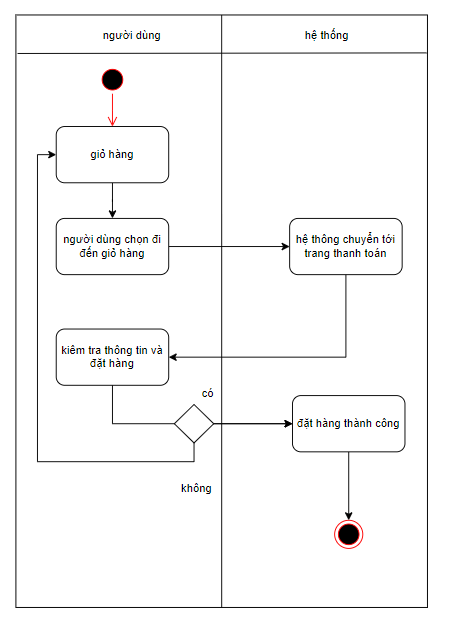
**5.2.6.Thêm sản phẩm vào giỏ hàng**

|  |  |
| --- | --- |
| * **Tên use case**: UC01\_Thêm sản phẩm vào giỏ hàng | |
| * **Mô tả sơ lược**: Người dùng thêm sản phẩm vào giỏ hàng | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. tại trang chủ |  |
| 1. người dùng chọn sản phẩm muốn mua | 1. hệ thông thông báo đã bỏ vào giỏ hàng |
|  |  |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  |  |



**5.2.7.Thanh toán sản phẩm**

|  |  |
| --- | --- |
| * **Tên use case**: UC01\_Thanh toán sản phẩm | |
| * **Mô tả sơ lược**: Người dùng thanh toán sản phẩm | |
| * **Actor chính**: Người dùng | |
| * **Actor phụ**: | |
| * **Tiền điều kiện (Pre-condition):** Người dùng đã có tài khoản | |
| * **Hậu điều kiện (Post-condition):** | |
| * **Luồng sự kiện chính (main flow):** | |
| **Actor** | **System** |
| 1. Gio hàng |  |
| 2. người dùng chọn đi đến giỏ hàng | 3. hệ thông chuyển tới trang thanh toán |
| 1. kiêm tra thông tin và đặt hàng | 1. đặt hàng thành công |
| * **Luồng sự kiện thay thế (alternate flow):** | |
|  | 4. quay lại giỏ hàng |



# 6. Mối quan hệ giữa kiến trúc và thiết kế phần mềm (Relationship between Software Architecture and Design)

1. Sự liên quan và phân biệt: Làm rõ sự khác biệt và mối liên hệ giữa kiến trúc phần mềm và thiết kế phần mềm.
2. Quy trình tích hợp: Cách hai lĩnh vực này tích hợp và hỗ trợ lẫn nhau trong quá trình phát triển phần mềm.
3. Xu hướng mới: Các xu hướng và công nghệ mới trong kiến trúc và thiết kế phần mềm, như kiến trúc serverless, AI trong thiết kế phần mềm, v.v.

# 7. Kết luận (Conclusion)

1. Tóm tắt: Tóm tắt lại những điểm chính đã trình bày trong bài tiểu luận.
2. Kết luận: Những kết luận quan trọng và ý nghĩa của nghiên cứu.
3. Hướng phát triển tương lai: Gợi ý về những nghiên cứu hoặc cải tiến có thể trong tương lai.

# 8. Tài liệu tham khảo (References)

Liệt kê tất cả các tài liệu, sách, bài báo và các nguồn khác mà bạn đã tham khảo để viết bài tiểu luận theo đúng định dạng trích dẫn yêu cầu (APA, MLA, Chicago, v.v.).