**BỘ GIÁO DỤC VÀ ĐẠO TẠO**

**TRƯỜNG ĐẠI HỌC NÔNG LÂM TPHCM**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KẾ HOẠCH**

**Phát triển phần mềm theo cách tiếp cận lặp và tăng**

**(Iterative and increment)**

**TÊN ĐỀ TÀI: PHẦN MỀM BÁN ĐIỆN THOẠI**

**Giáo viên hướng dẫn:** Thầy Lê Phi Hùng

**Sinh viên thực hiện:** Đoàn Nhất Linh : 21130091

Lê Thành Tâm : 21130526

CHƯƠNG I: CÁC YÊU CẦU CỦA ỨNG DỤNG

**1. Danh sách các user requirement của ứng dụng:**

* Giao diện đăng nhập.
* Tính năng tìm kiếm sản phẩm.
* Chức năng đăng ký tài khoản mới.
* Giao diện quản lý giỏ hàng.
* Tính năng đặt hàng và thanh toán.
* Tính năng gợi ý sản phẩm.
* Tính năng đánh giá và nhận xét sản phẩm.
* Tính năng tích hợp thanh toán qua ví điện tử.
* Tính năng chia sẻ sản phẩm lên mạng xã hội.

2. Sắp xếp thứ tự ưu các yêu cầu:

## 2.1 Phương pháp MoSCoW:

Phương pháp MoSCoW phân loại các yêu cầu thành các nhóm dựa trên mức độ ưu tiên:

* **Must have (Phải có)**: Đây là những yêu cầu bắt buộc phải có để sản phẩm hoặc dự án được coi là hoàn thành và chấp nhận được. Nếu thiếu bất kỳ yêu cầu nào trong danh sách này, sản phẩm sẽ không thể sử dụng được.
* **Should have (Nên có)**: Đây là những yêu cầu quan trọng nhưng không nhất thiết phải có trong phiên bản ban đầu. Nếu có thể triển khai được, các yêu cầu này sẽ cải thiện giá trị của sản phẩm.
* **Could have (Có thể có)**: Đây là những yêu cầu mà có thể thêm vào nếu thời gian và nguồn lực cho phép. Chúng không ảnh hưởng đến việc sản phẩm được coi là hoàn thành, nhưng có thể tăng thêm giá trị nếu triển khai được.
* **Won't have (Sẽ không có)**: Đây là những yêu cầu mà sẽ không được triển khai trong phiên bản hiện tại hoặc trong tương lai. Chúng có thể được xem xét trong các phiên bản sau hoặc được loại bỏ hoàn toàn.

## 2.2. Sắp xếp thứ tự ưu các yêu cầu

Dưới đây là một ví dụ về cách sắp xếp thứ tự ưu các yêu cầu bằng phương pháp MoSCoW:

1. **Must have**:
   * Giao diện đăng nhập.
   * Chức năng quản lý tài khoản người dùng.
   * Tính năng tìm kiếm sản phẩm.
2. **Should have**:
   * Chức năng đăng ký tài khoản mới.
   * Giao diện quản lý giỏ hàng.
   * Tính năng đặt hàng và thanh toán.
3. **Could have**:
   * Tính năng gợi ý sản phẩm.
   * Tính năng đánh giá và nhận xét sản phẩm.
4. **Won't have**:
   * Tính năng tích hợp thanh toán qua ví điện tử.
   * Tính năng chia sẻ sản phẩm lên mạng xã hội.

Bằng cách sắp xếp các yêu cầu theo phương pháp MoSCoW, chúng ta có thể đảm bảo rằng các yêu cầu quan trọng nhất và cần thiết nhất được ưu tiên và triển khai trước, trong khi vẫn giữ lại khả năng linh hoạt để thêm các tính năng phụ sau này nếu cần.

Chương 2: Lập kế hoạch thực hiện

1. Lựa chọn các yêu cầu đưa vào các phần gia tăng (Increment) phiên bản phần mềm sẽ phát triển:

Dựa trên sắp xếp thứ tự ưu các yêu cầu theo phương pháp MoSCoW và cách tiếp cận lặp và tăng, chúng ta có thể lựa chọn các yêu cầu để đưa vào các phần gia tăng (Increment) cho phiên bản phần mềm sẽ phát triển. Dưới đây là cách lựa chọn và lập kế hoạch thực hiện:

Phiên bản 1.0:

Increment 1:

Must have:

* Giao diện đăng nhập, trang chủ.
* Chức năng quản lý tài khoản người dùng.
* Tính năng tìm kiếm sản phẩm.

Phiên bản 1.1:

Increment 2:

Must have:

* Chức năng đăng ký tài khoản mới.
* Giao diện quản lý giỏ hàng.

Should have:

* Tính năng đặt hàng và thanh toán.

Phiên bản 1.2:

Increment 3:

Should have:

* Tính năng gợi ý sản phẩm.
* Tính năng đánh giá và nhận xét sản phẩm.

2.Kế hoạch thực hiện:

1. **Lập kế hoạch cho mỗi Increment:** Đặt ra mục tiêu cụ thể cho mỗi Increment, xác định nguồn lực cần thiết, và lập kế hoạch thời gian để hoàn thành.
2. **Phân công công việc:** Phân công các nhiệm vụ cụ thể cho các thành viên trong nhóm phát triển, dựa trên yêu cầu của từng Increment.
3. **Phát triển và kiểm thử:** Tiến hành phát triển các tính năng và giao diện mới cho mỗi Increment, sau đó thực hiện kiểm thử để đảm bảo tính ổn định và chất lượng.
4. **Tích hợp và triển khai:** Khi các tính năng đã được hoàn thành và kiểm thử kỹ lưỡng, tích hợp chúng vào phiên bản phần mềm chính và triển khai cho người dùng sử dụng.
5. **Thu thập phản hồi và cải thiện:** Thu thập phản hồi từ người dùng về các tính năng mới và dùng để cải thiện cho các phiên bản sau.

🡺Bằng cách này, chúng ta có thể phát triển phần mềm một cách có tổ chức, từng bước một, và đảm bảo rằng các tính năng quan trọng nhất được triển khai trước, trong khi vẫn giữ lại sự linh hoạt để thêm tính năng mới trong các phiên bản sau.

3.Lập kế hoạch thực hiện cho các phần gia tăng:

**3.1.Phần gia tăng 1:**

Dưới đây là một ví dụ về phân công các thành viên tham gia phát triển Increment 1 của Phiên bản 1.0 với các yêu cầu Must have đã nêu ở chương 1 phần 1.3:

**Increment 1: Phiên bản 1.0**

* **Mục tiêu**: Xây dựng một phiên bản cơ bản với các tính năng cơ bản như đăng nhập, quản lý tài khoản và tìm kiếm sản phẩm.

1. **Giao diện đăng nhập, trang chủ**:
   * **Thiết kế giao diện**: Thành viên A sẽ chịu trách nhiệm thiết kế giao diện người dùng cho trang đăng nhập.
   * **Phát triển và kiểm thử**: Thành viên B sẽ phát triển chức năng đăng nhập và thực hiện kiểm thử để đảm bảo tính ổn định và bảo mật.
   * **Thời gian hoàn thành**: 1 tuần.
2. **Chức năng quản lý tài khoản người dùng**:
   * **Thiết kế giao diện**: Thành viên C sẽ chịu trách nhiệm thiết kế giao diện người dùng cho trang quản lý tài khoản.
   * **Phát triển và kiểm thử**: Thành viên D sẽ phát triển chức năng quản lý tài khoản và thực hiện kiểm thử để đảm bảo tính ổn định và bảo mật.
   * **Thời gian hoàn thành**: 1 tuần.
3. **Tính năng tìm kiếm sản phẩm**:
   * **Thiết kế giao diện**: Thành viên E sẽ chịu trách nhiệm thiết kế giao diện người dùng cho trang tìm kiếm sản phẩm.
   * **Phát triển và kiểm thử**: Thành viên F sẽ phát triển chức năng tìm kiếm sản phẩm và thực hiện kiểm thử để đảm bảo tính ổn định và hiệu suất.
   * **Thời gian hoàn thành**: 1 tuần.

**3.2.Phần gia tăng 2:**

Dưới đây là một ví dụ về phân công các thành viên tham gia phát triển Increment 2 của Phiên bản 1.1 với các yêu cầu Must have và Should have đã nêu:

**Increment 2: Phiên bản 1.1**

* **Mục tiêu**: Bổ sung các tính năng mới như đăng ký tài khoản mới, quản lý giỏ hàng và tính năng đặt hàng và thanh toán.

1. **Chức năng đăng ký tài khoản mới (Must have)**:
   * **Thiết kế giao diện**: Thành viên A sẽ thiết kế giao diện cho trang đăng ký tài khoản mới.
   * **Phát triển và kiểm thử**: Thành viên B sẽ phát triển chức năng đăng ký tài khoản mới và thực hiện kiểm thử để đảm bảo tính ổn định và bảo mật.
2. **Giao diện quản lý giỏ hàng (Must have)**:
   * **Thiết kế giao diện**: Thành viên C sẽ thiết kế giao diện cho trang quản lý giỏ hàng.
   * **Phát triển và kiểm thử**: Thành viên D sẽ phát triển chức năng quản lý giỏ hàng và thực hiện kiểm thử để đảm bảo tính ổn định và hiệu suất.
3. **Tính năng đặt hàng và thanh toán (Should have)**:
   * **Thiết kế giao diện và logic**: Thành viên E sẽ thiết kế giao diện và logic cho tính năng đặt hàng và thanh toán.
   * **Phát triển và kiểm thử**: Thành viên F sẽ phát triển tính năng đặt hàng và thanh toán, và thực hiện kiểm thử để đảm bảo tính ổn định và hiệu suất.

**3.3.Phần gia tăng 3:**

Dưới đây là một ví dụ về phân công các thành viên tham gia phát triển Increment 3 của Phiên bản 1.2 với các yêu cầu Should have đã nêu:

**Increment 3: Phiên bản 1.2**

* **Mục tiêu**: Bổ sung các tính năng gợi ý sản phẩm và đánh giá, nhận xét sản phẩm để cải thiện trải nghiệm mua sắm của người dùng.

1. **Tính năng gợi ý sản phẩm**:
   * **Thiết kế giao diện và logic**: Thành viên A sẽ thiết kế giao diện và logic cho tính năng gợi ý sản phẩm, bao gồm việc xác định các thuật toán phù hợp để đề xuất các sản phẩm tương tự hoặc phù hợp với sở thích của người dùng.
   * **Phát triển và kiểm thử**: Thành viên B sẽ phát triển tính năng gợi ý sản phẩm và thực hiện kiểm thử để đảm bảo tính ổn định và hiệu suất.
2. **Tính năng đánh giá và nhận xét sản phẩm**:
   * **Thiết kế giao diện và logic**: Thành viên C sẽ thiết kế giao diện và logic cho tính năng đánh giá và nhận xét sản phẩm, bao gồm cơ chế cho phép người dùng đăng nhập và gửi đánh giá, nhận xét cho sản phẩm.
   * **Phát triển và kiểm thử**: Thành viên D sẽ phát triển tính năng đánh giá và nhận xét sản phẩm và thực hiện kiểm thử để đảm bảo tính ổn định và bảo mật.

**3.4.Thiết kế database:**

-Các thành viên làm chung.

Chương 3 Chọn một yêu cầu chức năng phần mềm.

# Chức năng quản lý giỏ hàng.

* 1. **Mô tả vấn đề:**

Tính năng giỏ hàng là một phần quan trọng của hệ thống thương mại điện tử, cung cấp trải nghiệm mua sắm linh hoạt và thuận tiện cho người dùng. Đây là một yêu cầu chức năng quan trọng và phổ biến, đặc biệt trong các ứng dụng thương mại điện tử. Dưới đây là một mô tả chi tiết về tính năng giỏ hàng và các tính năng liên quan mà bạn có thể cân nhắc:

\*\*Tính năng giỏ hàng:\*\*

1. \*\*Thêm sản phẩm vào giỏ hàng:\*\* Người dùng có thể thêm các sản phẩm mong muốn vào giỏ hàng từ danh sách sản phẩm. Có thể có các tùy chọn như chọn số lượng sản phẩm và chọn các biến thể (nếu có) như màu sắc, kích cỡ, v.v.

2. \*\*Hiển thị thông tin giỏ hàng:\*\* Người dùng có thể xem thông tin về sản phẩm đã thêm vào giỏ hàng, bao gồm hình ảnh sản phẩm, tên sản phẩm, số lượng, giá tiền, và tổng cộng.

3. \*\*Quản lý giỏ hàng:\*\* Người dùng có thể thêm, sửa đổi hoặc xóa sản phẩm từ giỏ hàng trước khi tiến hành thanh toán. Họ cũng có thể cập nhật số lượng sản phẩm trong giỏ hàng.

4. \*\*Tính năng tính tổng giá trị đơn hàng:\*\* Hệ thống sẽ tính tổng giá trị của các sản phẩm trong giỏ hàng dựa trên số lượng và giá tiền của mỗi sản phẩm, và hiển thị tổng giá trị này cho người dùng.

5. \*\*Tính năng tính phí vận chuyển (nếu có):\*\* Nếu có phí vận chuyển, hệ thống sẽ tính toán và hiển thị phí vận chuyển cho người dùng dựa trên địa chỉ giao hàng và các yêu cầu vận chuyển khác.

6. \*\*Liên kết với tính năng đặt hàng và thanh toán:\*\* Giỏ hàng sẽ được liên kết chặt chẽ với quy trình đặt hàng và thanh toán, để người dùng có thể dễ dàng chuyển từ việc xem giỏ hàng sang việc hoàn tất đơn hàng.

🡺Tính năng giỏ hàng sẽ cung cấp cho người dùng một cách tiện lợi và linh hoạt để lựa chọn và quản lý các sản phẩm mà họ muốn mua, tạo ra một trải nghiệm mua sắm trực tuyến tốt hơn.

**2.Thiết kế prototype:**  
**3. Thiết kế một giải pháp cho chức năng giỏ hàng dùng tiếp  
cận thiết kế phần mềm hướng đối tượng, mô-đun hóa:**

Để thiết kế một giải pháp cho chức năng giỏ hàng dùng tiếp cận thiết kế phần mềm hướng đối tượng và mô-đun hóa, chúng ta có thể sử dụng các khái niệm của hướng đối tượng như lớp, đối tượng, kế thừa, và mô-đun hóa để tổ chức và quản lý mã nguồn một cách hiệu quả. Dưới đây là một phác thảo về giải pháp:

**\*\*1. Lớp Product:\*\***

- \*\*Mô tả:\*\* Lớp này đại diện cho một sản phẩm trong giỏ hàng.

- \*\*Thuộc tính:\*\*

- id: ID của sản phẩm.

- name: Tên của sản phẩm.

- price: Giá của sản phẩm.

- quantity: Số lượng của sản phẩm.

- \*\*Phương thức:\*\*

- getPrice(): Trả về giá của sản phẩm.

- increaseQuantity(): Tăng số lượng của sản phẩm.

- decreaseQuantity(): Giảm số lượng của sản phẩm.

**\*\*2. Lớp Cart:\*\***

- \*\*Mô tả:\*\* Lớp này đại diện cho giỏ hàng.

- \*\*Thuộc tính:\*\*

- products: Một danh sách các sản phẩm trong giỏ hàng.

- \*\*Phương thức:\*\*

- addProduct(product): Thêm một sản phẩm vào giỏ hàng.

- removeProduct(product): Xóa một sản phẩm khỏi giỏ hàng.

**\*\*3. Lớp User:\*\***

- \*\*Mô tả:\*\* Lớp này đại diện cho người dùng.

- \*\*Thuộc tính:\*\*

- id: ID của người dùng.

- username: Tên đăng nhập của người dùng.

- email: Địa chỉ email của người dùng.

- cart: Giỏ hàng của người dùng.

- \*\*Phương thức:\*\*

- login(email, password): Phương thức để đăng nhập.

- logout(): Phương thức để đăng xuất.

- addToCart(product): Phương thức để thêm sản phẩm vào giỏ hàng.

- removeFromCart(product): Phương thức để xóa sản phẩm khỏi giỏ hàng.

- checkout(): Phương thức để tiến hành thanh toán.

\*\*4. Module Integration:\*\*

- Sử dụng các module nhỏ để thực hiện các chức năng cụ thể, chẳng hạn:

- Module đối tượng người dùng: Quản lý thông tin người dùng và quyền truy cập.

- Module giỏ hàng: Quản lý các sản phẩm trong giỏ hàng và tính toán tổng giá trị.

- Module thanh toán: Xử lý các giao dịch thanh toán và cập nhật trạng thái đơn hàng.

- Module sản phẩm: Quản lý thông tin sản phẩm và cung cấp các chức năng liên quan.

🡺Bằng cách sử dụng hướng đối tượng và mô-đun hóa, giải pháp này giúp chia nhỏ các chức năng thành các đối tượng và module riêng lẻ, giúp mã nguồn trở nên dễ quản lý, mở rộng và tái sử dụng.

# **4. Lược đồ lớp, lược đồ sequence, hoặc các thành phần thiết kế có liên quan khác:**