CÂU HỎI TRẮC NGHIỆM

Câu hỏi 1: Kỹ thuật nào sau đây thường được sử dụng để thu thập yêu cầu khách hàng?

A. Lập trình theo cặp

B. Phỏng vấn khách hàng

C. Kiểm thử đơn vị

D. Triển khai hệ thống

Câu hỏi 2: Trong bước tổng hợp kết quả yêu cầu, việc nào sau đây là cần thiết?

A. Xây dựng sơ đồ lớp

B. Phân loại yêu cầu và loại bỏ thông tin trùng lặp

C. Viết mã nguồn

D. Thiết kế giao diện

Câu hỏi 3: Use case mô tả:

A. Cách kiểm thử phần mềm

B. Cách người dùng tương tác với hệ thống

C. Cách quản lý tài liệu

D. Cách bảo trì phần mềm

Câu hỏi 4: Tại sao cần xây dựng danh sách từ khóa chuyên môn khi tìm hiểu lĩnh vực ứng dụng?

A. Để giảm thời gian lập trình

B. Để tăng tốc độ kiểm thử

C. Để đảm bảo đội phát triển và khách hàng hiểu rõ các thuật ngữ

D. Để giảm chi phí phát triển

Câu hỏi 5: Quan hệ nào sau đây trong use case mô tả một use case có thể mở rộng thêm chức năng trong một số điều kiện nhất định?

A. Include

B. Extend

C. Generalization

D. Aggregation

Câu hỏi 6: Khi mô tả yêu cầu bằng ngôn ngữ tự nhiên, điều quan trọng nhất là gì?

A. Sử dụng nhiều thuật ngữ kỹ thuật

B. Viết thật ngắn gọn và không cần kiểm tra lại với khách hàng

C. Viết rõ ràng, dễ hiểu và tránh từ đa nghĩa

D. Chỉ tập trung vào các yêu cầu phi chức năng

Câu hỏi 7: Mục tiêu chính của việc trích các use case là gì?

A. Thiết kế giao diện người dùng

B. Mô tả các chức năng mà hệ thống cung cấp cho người dùng

C. Viết mã nguồn cho hệ thống

D. Lập kế hoạch kiểm thử

Câu hỏi 8: Quan hệ nào giữa các use case thể hiện việc một use case phải gọi một use case khác để thực hiện chức năng đầy đủ?

A. Include

B. Extend

C. Generalization

D. Dependency

Câu hỏi 9: Hoạt động nào sau đây là bước đầu tiên trong việc xây dựng mô hình nghiệp vụ?

A. Thiết kế giao diện người dùng

B. Trích các use case

C. Viết mã nguồn

D. Thực hiện kiểm thử hệ thống

Câu hỏi 10: Một yêu cầu chức năng là gì?

A. Một yêu cầu mô tả cách hệ thống xử lý dữ liệu

B. Một yêu cầu về tốc độ xử lý của hệ thống

C. Một yêu cầu về khả năng bảo trì phần mềm

D. Một yêu cầu về giao diện người dùng

CÂU HỎI TRẢ LỜI NGẮN

1.Kỹ thuật phỏng vấn khách hàng là gì?

-Mục tiêu: Tìm hiểu và ghi nhận chi tiết những yêu cầu của khách hàng.

❑ Các kỹ thuật phỏng vấn:

-Phỏng vấn không cấu trúc: Khách hàng tự do trình bày các yêu cầu.

-Phỏng vấn có cấu trúc: Câu hỏi được chuẩn bị trước và khách hàng trả lời theo các chủ đề rõ ràng.

-Phỏng vấn bán cấu trúc: Kết hợp cả hai loại trên, vừa đặt câu hỏi theo trình tự, vừa tạo không gian để khách hàng bổ sung thông tin.

❑ Lưu ý khi phỏng vấn:

-Lắng nghe kỹ và ghi chú đầy đủ.

-Đặt câu hỏi rõ ràng, tránh gây hiểu lầm.

-Xác nhận lại với khách hàng để đảm bảo thông tin đúng.

2.Tại sao cần tổng hợp kết quả yêu cầu?

Tổng hợp kết quả yêu cầu là bước quan trọng giúp đảm bảo thông tin thu thập từ khách hàng được chính xác, rõ ràng và đầy đủ trước khi chuyển sang các giai đoạn tiếp theo như phân tích, thiết kế và phát triển hệ thống.

Lý do cần tổng hợp kết quả yêu cầu

1.Loại bỏ thông tin dư thừa, trùng lặp

- Khi thu thập yêu cầu từ nhiều nguồn (khách hàng, người dùng, tài liệu cũ...), có thể xuất hiện những yêu cầu trùng lặp hoặc mâu thuẫn.
- Việc tổng hợp giúp loại bỏ các yêu cầu trùng nhau và giữ lại thông tin quan trọng nhất.

2.Chuẩn hóa thông tin để dễ hiểu hơn

- Khách hàng thường mô tả yêu cầu theo cách riêng của họ, có thể sử dụng thuật ngữ chuyên môn hoặc diễn đạt không rõ ràng.

- Khi tổng hợp, nhóm phân tích hệ thống cần chuẩn hóa ngôn ngữ để đảm bảo tất cả các bên đều hiểu giống nhau.

3.Phân loại yêu cầu để dễ quản lý

- Yêu cầu có thể chia thành yêu cầu chức năng (hệ thống làm gì) và yêu cầu phi chức năng (hiệu suất, bảo mật...).
- Việc phân loại giúp nhóm phát triển dễ dàng tổ chức và ưu tiên các công việc quan trọng.

4.Giảm thiểu sai sót và mâu thuẫn trong yêu cầu

- Nếu không tổng hợp, các yêu cầu có thể bị mâu thuẫn nhau, dẫn đến hiểu lầm khi thiết kế và phát triển.
- Việc kiểm tra và xác nhận lại với khách hàng giúp phát hiện và giải quyết mâu thuẫn sớm.

5.Tạo tài liệu yêu cầu rõ ràng để làm cơ sở cho các giai đoạn tiếp theo

- Một tài liệu yêu cầu tốt giúp nhóm phát triển có hướng đi đúng đắn, tránh việc phải thay đổi quá nhiều trong quá trình phát triển.
- Nếu yêu cầu không rõ ràng ngay từ đầu, có thể dẫn đến phát triển sai hướng, gây lãng phí thời gian và chi phí.

Vai trò của mô tả bằng ngôn ngữ tự nhiên trong tổng hợp yêu cầu:

Lợi ích của việc sử dụng ngôn ngữ tự nhiên:

- Dễ hiểu: Cả khách hàng lẫn đội phát triển đều có thể đọc và hiểu.
- Không yêu cầu kỹ năng đặc biệt: Không cần phải hiểu UML hay các mô hình kỹ thuật.
- Dễ kiểm tra và xác nhận: Khách hàng có thể đọc và phản hồi nếu có sai sót.

Hạn chế của ngôn ngữ tự nhiên:

- Có thể gây hiểu nhầm: Cùng một câu có thể được hiểu theo nhiều cách khác nhau.
- Không đủ chính xác: Không mô tả được tất cả các chi tiết kỹ thuật.

-Giải pháp: Khi sử dụng ngôn ngữ tự nhiên để mô tả yêu cầu, cần viết rõ ràng, dễ hiểu, tránh từ ngữ mơ hồ và có thể kết hợp với bảng biểu, sơ đồ minh họa để tăng độ chính xác.

3.Use case là gì?

- Use Case (Trường hợp sử dụng) là một kỹ thuật được sử dụng trong phân tích và thiết kế phần mềm để mô tả cách người dùng tương tác với hệ thống nhằm đạt được một mục tiêu cụ thể.

Định nghĩa Use Case:

-Một Use Case mô tả một tập hợp các hành động mà hệ thống thực hiện để cung cấp giá trị cho người dùng (Actor). Nó giúp xác định chức năng của hệ thống và mối quan hệ giữa các thành phần trong hệ thống.

-Ví dụ đơn giản về Use Case:
Trong một hệ thống đặt vé máy bay, người dùng có thể thực hiện các hành động như:

- Tìm kiếm chuyến bay
- Đặt vé
- Thanh toán
- Hủy vé

-Mỗi hành động trên là một Use Case vì nó mô tả một chức năng mà hệ thống cung cấp cho người dùng.

Các thành phần chính của một Use Case:

1. Tên Use Case: Diễn tả hành động chính, ví dụ: *"Đăng nhập vào hệ thống"*.
2. Actor (Tác nhân): Người dùng hoặc hệ thống khác tương tác với hệ thống. Ví dụ: *Khách hàng, Nhân viên, Hệ thống thanh toán*.
3. Mô tả: Tóm tắt về mục tiêu của Use Case.
4. Luồng sự kiện chính (Main Flow): Trình tự các bước mà người dùng thực hiện để hoàn thành Use Case.
5. Luồng thay thế (Alternative Flow): Các trường hợp đặc biệt hoặc ngoại lệ, như *"Sai mật khẩu khi đăng nhập"*.
6. Điều kiện tiên quyết: Những điều kiện cần có trước khi Use Case có thể xảy ra.
7. Kết quả mong đợi: Hệ thống phải phản hồi như thế nào sau khi Use Case kết thúc.

Biểu đồ Use Case (Use Case Diagram)

- Biểu đồ Use Case là một công cụ giúp minh họa mối quan hệ giữa Actor và các Use Case của hệ thống bằng hình ảnh trực quan.

- Ví dụ về biểu đồ Use Case:

Hệ thống ATM có các Use Case sau:

- Rút tiền
- Nạp tiền
- Kiểm tra số dư

Biểu đồ Use Case cho hệ thống ATM sẽ có:

- Actor: Người dùng (Khách hàng)
- Use Case: "Rút tiền", "Nạp tiền", "Kiểm tra số dư"
- Mối quan hệ: Khách hàng tương tác với hệ thống để thực hiện các Use Case này.

4.Mục tiêu của việc xây dựng danh sách từ khóa chuyên môn là gì?

-Mục tiêu: Giúp đội phát triển hiểu rõ các thuật ngữ chuyên ngành mà khách hàng sử dụng.

5.Quan hệ Include giữa các use case là gì?

- Định nghĩa:

Quan hệ Include trong Use Case là mối quan hệ bắt buộc giữa các Use Case, trong đó một Use Case luôn luôn gọi một Use Case khác để hoàn thành chức năng của nó.

- Tóm tắt:

- Use Case A (*gốc*) bao gồm (Include) Use Case B (*phụ*).
- Use Case B luôn luôn được thực thi khi Use Case A được thực thi.
- Dùng để tránh lặp lại các chức năng dùng chung giữa nhiều Use Case khác nhau.

6.Quan hệ Extend giữa các use case là gì?

- Định nghĩa:

Quan hệ Extend trong Use Case là mối quan hệ mở rộng, trong đó một Use Case có thể mở rộng một Use Case khác trong một số điều kiện nhất định, nhưng không bắt buộc phải xảy ra.

- Tóm tắt:

- Use Case A (*gốc*) có thể được mở rộng bởi Use Case B (*mở rộng*).

- Use Case B chỉ được thực hiện khi có điều kiện cụ thể.
- Dùng để bổ sung tính năng tùy chọn cho Use Case A mà không làm thay đổi logic chính.

7.Yêu cầu phi chức năng là gì?

-Định nghĩa:

Yêu cầu phi chức năng (Non-functional Requirements - NFRs) là các yêu cầu không liên quan trực tiếp đến chức năng của hệ thống, mà mô tả cách hệ thống hoạt động hơn là hệ thống làm gì.

- Tóm lại:

- Không liên quan đến tính năng cụ thể của phần mềm.
- Mô tả các đặc tính về hiệu suất, bảo mật, khả năng mở rộng,...
- Ảnh hưởng đến trải nghiệm người dùng và hiệu quả của hệ thống.

- Ví dụ về yêu cầu phi chức năng:

Dưới đây là một số loại yêu cầu phi chức năng phổ biến:

Loại yêu cầu	Ví dụ
Hiệu suất (Performance)	Hệ thống phải xử lý 1000 yêu cầu mỗi giây.
Bảo mật (Security)	Dữ liệu người dùng phải được mã hóa AES-256.
Khả năng mở rộng (Scalability)	Hệ thống phải hỗ trợ 1 triệu người dùng đồng thời.
Khả dụng (Availability)	Hệ thống phải hoạt động 99.99% thời gian.
Dễ sử dụng (Usability)	Người dùng mới có thể học cách sử dụng trong vòng 5 phút.
Khả năng bảo trì (Maintainability)	Hệ thống phải dễ dàng cập nhật mà không ảnh hưởng đến dịch vụ.
Khả năng tương thích (Compatibility)	Ứng dụng phải chạy trên Windows, Mac, và Linux.

8.Mô hình nghiệp vụ là gì?

- Định nghĩa:

Mô hình nghiệp vụ (Business Model trong phần mềm) là bản mô tả cách một tổ chức hoặc hệ thống phần mềm hoạt động để cung cấp giá trị, bao gồm các quy trình, tác nhân, quy tắc kinh doanh và cách dữ liệu luân chuyển.

- Mục đích chính:

Giúp phân tích, hiểu rõ nhu cầu và quy trình làm việc của doanh nghiệp.
Là cơ sở để thiết kế hệ thống phần mềm phù hợp với yêu cầu thực tế.
Hỗ trợ tối ưu hóa hoạt động kinh doanh bằng công nghệ.

- Các thành phần của mô hình nghiệp vụ trong phần mềm:

Thành phần	Mô tả
Tác nhân	Người hoặc hệ thống tham gia vào quy trình (Người dùng, Quản trị viên, Hệ thống bên thứ ba,...)
Use Case	Các chức năng chính mà hệ thống phần mềm cung cấp.
Quy trình nghiệp vụ (Business Process)	Các bước thực hiện công việc trong hệ thống phần mềm.
Luồng dữ liệu (Data Flow)	Cách thông tin được nhập, xử lý và xuất ra trong hệ thống.
Quy tắc kinh doanh (Business Rules)	Các quy định, chính sách chi phối hoạt động của hệ thống.

-Ví dụ về mô hình nghiệp vụ trong phần mềm

Ví dụ 1: Hệ thống bán hàng online

Tác nhân: Khách hàng, Nhân viên kho, Nhân viên giao hàng.
Quy trình nghiệp vụ:
1.Khách hàng đặt hàng trên website.
2.Hệ thống kiểm tra kho hàng.
3.Nhân viên kho xác nhận đơn hàng.
4.Nhân viên giao hàng vận chuyển.
5.Khách nhận hàng và thanh toán.

Quy tắc kinh doanh:

- Nếu số lượng hàng trong kho < 5, hiển thị cảnh báo hết hàng.
- Đơn hàng trên 1 triệu VNĐ được giảm giá 5%.

Ví dụ 2: Hệ thống ngân hàng trực tuyến

Tác nhân: Khách hàng, Nhân viên ngân hàng, Hệ thống ATM.
Quy trình nghiệp vụ:
1.Khách hàng đăng nhập vào ứng dụng ngân hàng.
2.Thực hiện giao dịch (chuyển tiền, rút tiền, thanh toán hoá đơn).
3.Hệ thống kiểm tra số dư.
4.Nếu hợp lệ, giao dịch được xác nhận và cập nhật số dư.

-Lợi ích của mô hình nghiệp vụ trong phần mềm

Giúp phân tích và tối ưu hóa quy trình nghiệp vụ
Hỗ trợ thiết kế hệ thống phần mềm đúng yêu cầu
Tăng hiệu quả quản lý và tự động hóa công việc
Giúp dễ dàng bảo trì, nâng cấp hệ thống

9.Điều kiện trước và điều kiện sau của use case là gì?

- Định nghĩa Use Case:

Use Case mô tả **các chức năng của hệ thống** từ góc nhìn của người dùng, thể hiện cách hệ thống tương tác với tác nhân (Actor).

Trong một **Use Case**, để đảm bảo tính logic và đúng trình tự, cần xác định:

- **Điều kiện trước (Precondition)**
- **Điều kiện sau (Postcondition)**

- Điều kiện trước của một Use Case là các điều kiện hoặc trạng thái mà hệ thống phải thỏa mãn trước khi Use Case có thể được thực hiện.

Mục đích:
Đảm bảo hệ thống có đủ thông tin và trạng thái phù hợp để thực hiện Use Case.
Ngăn chặn các trường hợp lỗi khi điều kiện chưa sẵn sàng.

Ví dụ:

- **Use Case: Đăng nhập vào hệ thống**
    - Điều kiện trước:
        - Người dùng đã có tài khoản hợp lệ.
        - Kết nối mạng ổn định.

- **Use Case: Thanh toán giỏ hàng**
    - Điều kiện trước:
        - Giỏ hàng không rỗng.
        - Người dùng đã nhập thông tin thanh toán.

- **Điều kiện sau** của một Use Case là trạng thái của hệ thống sau khi Use Case được thực thi thành công.

Mục đích:
Đảm bảo hệ thống hoạt động đúng theo quy trình.
Xác định những thay đổi xảy ra sau khi Use Case hoàn thành.

Ví dụ:

- **Use Case: Đăng nhập vào hệ thống**
    - Điều kiện sau:
        - Người dùng được chuyển đến trang chính.
        - Hệ thống lưu lại thông tin đăng nhập nếu có chọn “Ghi nhớ đăng nhập”.
- **Use Case: Thanh toán giỏ hàng**
    - Điều kiện sau:
        - Số dư tài khoản của người dùng bị trừ.
        - Hệ thống cập nhật trạng thái đơn hàng thành "Đã thanh toán".

Tại sao điều kiện trước và điều kiện sau quan trọng?

Giúp hệ thống hoạt động đúng logic.
Tránh lỗi khi chạy Use Case.
Đảm bảo tính toàn vẹn của dữ liệu.
Giúp lập trình viên và tester xác định các trường hợp kiểm thử chính xác.

10.Một số lưu ý khi trích các use case là gì?

- Một số lưu ý khi trích các use case:

+Đảm bảo mỗi use case thể hiện một chức năng rõ ràng.

+Tránh viết quá chi tiết hoặc quá chung chung.

+Kiểm tra sự nhất quán giữa các use case.

CÂU HỎI TÌNH HUỐNG

Câu 1. Khách hàng không hiểu rõ các thuật ngữ kỹ thuật trong tài liệu yêu cầu. Là trưởng nhóm phát triển, bạn sẽ làm gì?

Là trưởng nhóm phát triển, bạn có thể thực hiện các biện pháp sau để giúp khách hàng hiểu rõ hơn:

- **Sử dụng ngôn ngữ đơn giản, dễ hiểu**: Tránh thuật ngữ chuyên môn hoặc giải thích rõ ràng bằng cách sử dụng ví dụ thực tế.
- **Xây dựng danh sách thuật ngữ chuyên môn**: Tạo một bảng thuật ngữ để giải thích các khái niệm quan trọng.
- **Sử dụng hình ảnh và sơ đồ minh họa**: Sơ đồ use case, wireframe, mô hình luồng dữ liệu giúp khách hàng dễ hiểu hơn.
- **Trao đổi trực tiếp với khách hàng**: Thực hiện các buổi họp hoặc workshop để giải thích và đảm bảo họ hiểu đúng.
- **Sử dụng tài liệu mô tả theo cách tiếp cận từ góc nhìn của khách hàng**: Viết tài liệu yêu cầu dưới dạng **user stories** hoặc **use case** thay vì mô tả kỹ thuật phức tạp.

Câu 2. Trong quá trình lấy yêu cầu, khách hàng liên tục thay đổi ý kiến về chức năng cần thiết. Đội phát triển nên xử lý ra sao?

Việc khách hàng thay đổi yêu cầu là điều phổ biến, đặc biệt trong các dự án phần mềm. Để xử lý hiệu quả, nhóm phát triển có thể:

- **Sử dụng phương pháp phát triển linh hoạt (Agile)**: Agile (ví dụ: Scrum, Kanban) cho phép thay đổi yêu cầu linh hoạt theo từng sprint.
- **Xác định yêu cầu cốt lõi**: Tập trung vào các tính năng quan trọng trước, sau đó mới xem xét các yêu cầu thay đổi.
- **Ưu tiên yêu cầu**: Sử dụng **MoSCoW (Must-have, Should-have, Could-have, Won't-have)** để xác định mức độ quan trọng của từng yêu cầu.
- **Thiết lập quy trình quản lý thay đổi yêu cầu**: Yêu cầu thay đổi cần được đánh giá về tác động đến thời gian, chi phí và nguồn lực trước khi chấp nhận.
- **Giải thích chi phí của việc thay đổi**: Nếu yêu cầu thay đổi ảnh hưởng lớn đến tiến độ hoặc chi phí, cần trao đổi rõ ràng với khách hàng để họ cân nhắc lại.
- **Ghi nhận yêu cầu thay đổi và cập nhật tài liệu**: Luôn có một hệ thống quản lý yêu cầu để theo dõi các thay đổi và cập nhật tài liệu kịp thời.

Câu 3. Một dự án phần mềm quản lý bán hàng gặp vấn đề khi các yêu cầu được mô tả quá chung chung, khó thực hiện. Đội phát triển cần làm gì để khắc phục?

- Cần có cuộc họp lại với khách hàng để làm rõ các yêu cầu chưa rõ ràng.

- Tổng hợp lại yêu cầu của khách hàng loại bỏ mâu thuẫn trùng lặp và thống nhất lại khách hàng.
- Lập bảng mô tả chi tiết yêu cầu khách hàng theo ngôn ngữ tự nhiên.
- Sử dụng bảng mô tả từ khóa chuyên môn để đảm bảo nhóm phát triển hiểu rõ các chi tiết của yêu cầu.
- Sử dụng các công cụ để mô hình hóa như sơ đồ luồng, sơ đồ UML, ERD hoặc các mẫu thử,...
- Họp lại nhóm phát triển để đảm bảo mô tả yêu cầu được tất cả thành viên đều hiểu và hợp lý trước khi thực hiện dự án.

Câu 4. Khách hàng đưa ra yêu cầu không rõ ràng và chỉ có thể mô tả một cách sơ lược. Làm thế nào để thu thập yêu cầu đầy đủ từ khách hàng?

- Chuẩn bị các câu hỏi cần thiết dựa trên các mô tả trước đó của khách hàng để phục vụ cho quá trình phát triển và thực hiện phỏng vấn có cấu trúc lại với khách hàng, hỏi khách hàng theo những câu hỏi đã chuẩn bị cũng như đề xuất các hướng giải quyết nếu khách hàng không có ý tưởng gì.
- Có thể tạo trước các mẫu thử để trao đổi lại với khách hàng trên mẫu đó.
- Xác nhận lại với khách hàng về các yêu cầu mà nhóm đã thu thập được để đảm bảo đủ yêu cầu để phát triển dự án và đúng yêu cầu của khách hàng.

Câu 5. Trong quá trình xây dựng mô hình nghiệp vụ, một số use case bị trùng lặp về chức năng. Bạn sẽ xử lý tình huống này như thế nào?

- Phân tích lại use case để xem tại sao lại có sự trùng lặp do quá trình thiết kế use case có sai sót hay do mô tả thiết kế không rõ ràng.
- Xem xét lại các use case trùng lặp và tổng quát hóa use case đó lên cho các tác nhân sử dụng nếu các use case đó trùng lặp do mô tả quá chi tiết cho từng tác nhân.
- Mô tả chi tiết lại cho từng use case để đảm bảo các use case thực hiện một chức năng rõ ràng. Mô tả các tác nhân tham gia và dòng sự kiện chính của use case để đảm bảo các use case đều có nghiệp vụ riêng biệt và không bị trùng lặp về chức năng.

Câu 6. Đội phát triển và khách hàng có ý kiến khác nhau về ý nghĩa của một số từ khóa chuyên môn. Là trưởng dự án, bạn sẽ làm gì?

- Tạo cuộc họp giữa đội phát triển và khách hàng để làm rõ và thống nhất lại bảng các từ khóa chuyên môn.
- Nghe các định nghĩa khác nhau từ phía khách hàng và đội phát triển để tìm ra điểm mâu thuẫn và thảo luận về nó để có thể đưa ra ý nghĩa chính xác của từ khóa đó.
- Sử dụng ngôn ngữ dễ hiểu và thay thế các từ chuyên ngành phức tạp ít được sử dụng thành các từ có thể mô tả ý nghĩa đó nhưng được sử dụng rộng rãi hơn.
- Đưa ra các ví dụ liên quan để giải thích, minh họa về các khái niệm.

- Xây dựng lại danh sách các từ khóa chuyên môn và gửi cho khách hàng, đội phát triển và các bên liên quan để danh sách phải được sự thống nhất từ tất cả mọi người và thống nhất sử dụng trong cả quá trình phát triển.

Câu 7. Sau khi hoàn thành việc trích các use case, đội phát triển nhận ra rằng một số chức năng quan trọng bị bỏ sót. Hãy đề xuất cách giải quyết.

- Nếu sau khi hoàn thành việc trích các Use Case, đội phát triển nhận ra rằng một số chức năng quan trọng bị bỏ sót, họ có thể áp dụng các phương pháp sau để giải quyết:

1. Xem xét và cập nhật lại tập Use Case

Rà soát lại toàn bộ danh sách Use Case để kiểm tra xem những chức năng nào bị thiếu. Bổ sung các Use Case còn thiếu bằng cách:

- Thêm Use Case mới nếu đó là chức năng hoàn toàn mới.
- Mở rộng Use Case hiện có bằng quan hệ Extend hoặc Include, nếu chức năng bị thiếu là một phần của Use Case đã có.

Ví dụ:

- Nếu hệ thống đặt hàng online ban đầu chỉ có Use Case "Thanh toán", nhưng quên chức năng "Áp dụng mã giảm giá", thì có thể:
    - Thêm Use Case mới: *Áp dụng mã giảm giá*.
    - Sử dụng Include: Nếu mã giảm giá luôn phải được áp dụng trong quá trình thanh toán.
    - Sử dụng Extend: Nếu mã giảm giá là chức năng tùy chọn trong thanh toán.

2. Phỏng vấn lại khách hàng và các bên liên quan

- Gặp gỡ khách hàng hoặc người dùng để xác minh lại yêu cầu.
- Sử dụng các kỹ thuật thu thập yêu cầu như:

- Phỏng vấn khách hàng để tìm hiểu chức năng còn thiếu.
- Lập bảng câu hỏi để xác nhận với người dùng về các tình huống sử dụng.
- Quan sát thực tế sử dụng để tìm ra các chức năng bị bỏ sót.

3. Vẽ lại mô hình Use Case và kiểm tra sự nhất quán

- Kiểm tra mối quan hệ giữa các Use Case để đảm bảo không bỏ sót chức năng phụ thuộc.
- Sử dụng biểu đồ Use Case để dễ dàng nhìn thấy các chức năng bị thiếu.

4. Họp nhóm phát triển để rà soát lại yêu cầu

Đội phát triển nên tổ chức họp đánh giá lại yêu cầu để xem xét:

- Những chức năng nào bị thiếu.
- Chúng có ảnh hưởng đến các Use Case khác không.
- Cách bổ sung mà không làm ảnh hưởng lớn đến hệ thống.

5. Cập nhật tài liệu Use Case và kiểm thử lại

Sau khi bổ sung chức năng bị thiếu, cần cập nhật lại:

- Tài liệu mô tả Use Case.
- Kịch bản kiểm thử để đảm bảo chức năng mới hoạt động đúng.

- Kết luận:

Cách giải quyết tốt nhất là rà soát lại yêu cầu, bổ sung Use Case còn thiếu, xác nhận với khách hàng và cập nhật tài liệu.
Luôn kiểm tra kỹ trước khi hoàn thiện mô hình Use Case để tránh thiếu sót ngay từ đầu.

Câu 8. Trong quá trình xây dựng mô hình nghiệp vụ, khách hàng yêu cầu bổ sung thêm một số chức năng mới. Đội phát triển nên làm gì?

- Khi khách hàng yêu cầu thêm chức năng mới trong quá trình xây dựng mô hình nghiệp vụ, đội phát triển cần xử lý một cách có hệ thống để đảm bảo không ảnh hưởng tiêu cực đến tiến độ và chất lượng dự án. Dưới đây là các bước đề xuất:

1. Phân tích yêu cầu mới của khách hàng

- Tiếp nhận yêu cầu từ khách hàng và xác định chức năng mới có thực sự cần thiết không.
- Xác định xem yêu cầu này có ảnh hưởng đến các chức năng hiện tại không.
- Nếu có ảnh hưởng, cần đánh giá mức độ thay đổi cần thiết trong hệ thống.

➞ Câu hỏi quan trọng cần đặt ra:

- Chức năng mới này có phù hợp với mục tiêu ban đầu không?
- Nó có ảnh hưởng đến các Use Case hiện có không?
- Nó có làm thay đổi quy trình nghiệp vụ hiện tại không?

2. Trao đổi với khách hàng để làm rõ yêu cầu

- Làm rõ kỳ vọng của khách hàng: Chức năng mới cần hoạt động như thế nào?
- Ưu tiên hóa: Chức năng này quan trọng đến mức nào so với những gì đã có?
- Xác nhận phạm vi thay đổi: Có cần sửa đổi nhiều quy trình hay chỉ là bổ sung nhỏ?

3. Đánh giá tác động của yêu cầu mới

- Xác định chức năng mới có ảnh hưởng đến kiến trúc hệ thống không.
- Kiểm tra xem có cần thay đổi Use Case hiện tại hay bổ sung Use Case mới.
- Đánh giá tác động đến:

  - Thời gian hoàn thành dự án
  - Chi phí phát triển
  - Rủi ro và tính ổn định của hệ thống

➞ Nếu ảnh hưởng quá lớn, có thể cần thương lượng lại với khách hàng về phạm vi và thời gian phát triển.

4. Cập nhật mô hình nghiệp vụ

Nếu chức năng mới được chấp nhận, đội phát triển cần cập nhật:

- Mô hình Use Case: Bổ sung hoặc sửa đổi Use Case
- Sơ đồ quy trình nghiệp vụ (BPMN, Activity Diagram, v.v.) nếu có thay đổi về luồng nghiệp vụ
- Tài liệu đặc tả yêu cầu (SRS – Software Requirement Specification)

5. Xác định mức độ ưu tiên và cập nhật kế hoạch phát triển

Nếu chức năng mới quan trọng, cần điều chỉnh lại kế hoạch phát triển phần mềm.
Nếu chức năng không cấp bách, có thể đưa vào danh sách cải tiến cho phiên bản sau.
Xác định xem cần tăng tài nguyên (thêm nhân sự, thời gian, ngân sách) hay không.

6. Kiểm tra lại hệ thống sau khi bổ sung chức năng mới

Kiểm thử lại toàn bộ hệ thống để đảm bảo chức năng mới không gây lỗi.
Cập nhật kế hoạch kiểm thử để bao gồm các trường hợp kiểm thử cho chức năng mới.

Kết luận:

Đội phát triển không nên vội vàng chấp nhận yêu cầu mới mà cần phân tích kỹ lưỡng để đảm bảo tính khả thi và tránh rủi ro.

Nếu yêu cầu hợp lý, cần cập nhật mô hình nghiệp vụ, tài liệu và kế hoạch phát triển để đảm bảo sự nhất quán trong hệ thống.

Câu 9. Một nhóm phát triển gặp khó khăn trong việc mô tả chi tiết các use case vì chưa hiểu rõ quy trình nghiệp vụ. Hãy đề xuất giải pháp.

- Nếu nhóm phát triển chưa hiểu rõ quy trình nghiệp vụ, việc mô tả Use Case sẽ gặp nhiều sai sót, thiếu sót hoặc mơ hồ. Để giải quyết vấn đề này, nhóm cần thực hiện các bước sau:

1. Trao đổi trực tiếp với khách hàng và chuyên gia nghiệp vụ (SME - Subject Matter Expert)

- Tổ chức các buổi họp, phỏng vấn hoặc workshop với khách hàng hoặc chuyên gia nghiệp vụ để làm rõ quy trình hoạt động.
- Đặt câu hỏi chi tiết về các bước thực hiện trong quy trình, vai trò của người dùng, và tình huống đặc biệt.
- Sử dụng các câu hỏi mở để thu thập thông tin đầy đủ hơn.

Ví dụ câu hỏi:

- Người dùng sẽ thực hiện chức năng này theo trình tự nào?
- Có những trường hợp ngoại lệ nào cần xử lý không?
- Nếu một bước trong quy trình bị lỗi, hệ thống sẽ xử lý ra sao?

2. Nghiên cứu tài liệu nghiệp vụ có sẵn

- Yêu cầu khách hàng cung cấp quy trình làm việc hiện tại, sổ tay hướng dẫn, hoặc tài liệu mô tả quy trình nghiệp vụ nếu có.
- Nếu khách hàng chưa có tài liệu chính thức, nhóm có thể đề xuất viết lại quy trình dưới dạng sơ đồ nghiệp vụ (BPMN, Activity Diagram, v.v.).

3. Sử dụng phương pháp kịch bản (Scenario-Based Analysis)

- Viết các kịch bản thực tế mà người dùng sẽ trải qua khi sử dụng hệ thống.
- Mô tả chi tiết các bước thao tác, điều kiện đầu vào, đầu ra mong muốn, và tình huống ngoại lệ.
- Khi kịch bản đủ rõ ràng, có thể chuyển đổi thành Use Case chi tiết.

Ví dụ kịch bản:
Người dùng đăng nhập vào hệ thống:

1. Người dùng nhập tên đăng nhập và mật khẩu.

2. Hệ thống kiểm tra thông tin đăng nhập.
    - Nếu hợp lệ, hệ thống chuyển đến trang chính.
    - Nếu sai, hiển thị thông báo lỗi và yêu cầu nhập lại.

➞ Từ kịch bản này, có thể tạo Use Case "Đăng nhập hệ thống" với các bước chi tiết.

4. Sử dụng sơ đồ trực quan để làm rõ quy trình nghiệp vụ

- Sơ đồ Use Case: Xác định các tác nhân (actor) và các Use Case chính.
- Activity Diagram: Diễn giải chi tiết luồng nghiệp vụ.
- State Diagram: Nếu hệ thống có nhiều trạng thái, mô tả các trạng thái và cách chuyển đổi giữa chúng.

Ví dụ: Nếu nhóm phát triển gặp khó khăn khi mô tả Use Case "Quản lý đơn hàng", có thể vẽ sơ đồ hoạt động (Activity Diagram) để hiểu rõ quy trình từ khi khách đặt hàng đến khi đơn hàng được giao.

5. Xây dựng mô hình Use Case theo cách tiếp cận lặp (Iterative Approach)

- Bắt đầu từ Use Case tổng quát, sau đó dần dần bổ sung chi tiết dựa trên thông tin thu thập được.
- Sau mỗi lần cập nhật, kiểm tra lại với khách hàng để đảm bảo mô tả đúng yêu cầu thực tế.
- Tránh viết Use Case quá chi tiết ngay từ đầu vì có thể phải thay đổi nhiều lần.

6. Xác thực Use Case bằng cách mô phỏng hoặc thử nghiệm với khách hàng

- Dùng bản mô phỏng hoặc wireframe để giúp khách hàng dễ dàng hiểu và xác nhận Use Case.
- Thực hiện thử nghiệm trên một hệ thống mô phỏng để kiểm tra xem quy trình có hợp lý không.

Kết luận:

- Khi gặp khó khăn trong việc mô tả Use Case, quan trọng nhất là hiểu rõ quy trình nghiệp vụ trước.
- Cách tốt nhất là trao đổi với khách hàng, sử dụng sơ đồ trực quan, và xây dựng Use Case theo cách tiếp cận lặp.
- Nếu vẫn chưa rõ, có thể mô phỏng hoặc thử nghiệm để kiểm tra lại tính chính xác.

Câu 10. Khách hàng yêu cầu thêm một chức năng mới khi hệ thống đã bước vào giai đoạn thiết kế chi tiết. Đội phát triển cần xử lý như thế nào?

- Khi hệ thống đã bước vào giai đoạn thiết kế chi tiết, việc bổ sung chức năng mới có thể ảnh hưởng đến tiến độ, ngân sách và tính ổn định của hệ thống. Đội phát triển cần thực hiện các bước sau để đánh giá và xử lý yêu cầu một cách hợp lý.

1. Xác định và phân tích yêu cầu mới

- Trao đổi với khách hàng để hiểu rõ bản chất và phạm vi của chức năng mới.
- Xác định mức độ ưu tiên của yêu cầu:

- Yêu cầu bắt buộc phải có để hệ thống hoạt động đúng.
- Yêu cầu có thể trì hoãn sang giai đoạn tiếp theo.
- Kiểm tra xem yêu cầu mới có ảnh hưởng đến các chức năng hiện có hay không.

Ví dụ:

- Nếu khách hàng yêu cầu thêm tính năng xuất báo cáo, đội phát triển cần xác định:
    - Dữ liệu nào cần xuất?
    - Định dạng báo cáo ra sao?
    - Hệ thống hiện tại có hỗ trợ dữ liệu này không?

2. Đánh giá tác động của yêu cầu mới

- Ảnh hưởng đến kiến trúc hệ thống: Yêu cầu mới có làm thay đổi cấu trúc dữ liệu, logic xử lý hoặc giao diện không?
- Ảnh hưởng đến tiến độ và ngân sách: Cần thêm bao nhiêu thời gian và chi phí để thực hiện?
- Rủi ro kỹ thuật: Có gây xung đột với các chức năng đã thiết kế không?

Ví dụ:

- Nếu yêu cầu mới chỉ ảnh hưởng đến giao diện người dùng, việc triển khai có thể nhanh chóng.
- Nếu yêu cầu mới cần thay đổi mô hình dữ liệu, có thể mất nhiều thời gian để sửa đổi.

3. Đề xuất phương án xử lý

Dựa trên phân tích, nhóm có thể đề xuất các phương án sau:

Phương án 1: Chấp nhận và điều chỉnh kế hoạch

- Nếu yêu cầu cần thiết và có thể thực hiện mà không ảnh hưởng lớn, đội phát triển sẽ cập nhật tài liệu thiết kế và điều chỉnh tiến độ.

Phương án 2: Hoãn lại sang phiên bản tiếp theo

- Nếu yêu cầu không cấp thiết hoặc có ảnh hưởng lớn đến tiến độ, có thể thương lượng với khách hàng để bổ sung vào giai đoạn nâng cấp sau này.

Phương án 3: Đề xuất giải pháp thay thế

- Nếu việc bổ sung gây rủi ro lớn, có thể đề xuất một giải pháp đơn giản hơn hoặc sử dụng chức năng hiện có để đáp ứng một phần yêu cầu.

Ví dụ:

- Khách hàng muốn thêm chức năng gửi email tự động cho từng đơn hàng.
    - Nếu thời gian gấp, có thể đề xuất xuất file báo cáo thay vì gửi email tự động.
    - Nếu thời gian cho phép, đội phát triển có thể triển khai tính năng này nhưng cần điều chỉnh tiến độ dự án.

4. Cập nhật tài liệu thiết kế và lộ trình phát triển

- Nếu chấp nhận yêu cầu, đội phát triển cần cập nhật tài liệu thiết kế, sơ đồ hệ thống, và lộ trình phát triển để phản ánh thay đổi.
- Nếu từ chối hoặc trì hoãn, cần ghi nhận yêu cầu vào backlog để xem xét trong các phiên bản tiếp theo.

Kết luận:

- Khi khách hàng yêu cầu thêm chức năng mới trong giai đoạn thiết kế chi tiết, đội phát triển cần đánh giá tác động, thương lượng với khách hàng và đưa ra phương án hợp lý. -- Nếu yêu cầu ảnh hưởng lớn, có thể hoãn lại hoặc thay thế bằng giải pháp đơn giản hơn để không làm chậm tiến độ.
- Nếu chấp nhận yêu cầu, cần cập nhật kế hoạch và tài liệu thiết kế để đảm bảo tính nhất quán của hệ thống.