Trắc Nghiệm

1. 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

2. 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

3. 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

4. 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

5. 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

6. 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

7. 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

8. 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

9. 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

10. 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?

Kỹ thuật phỏng vấn khách hàng là một phương pháp thu thập yêu cầu trong quá trình phát triển phần mềm hoặc hệ thống thông tin, trong đó nhóm phát triển trực tiếp đặt câu hỏi cho khách hàng hoặc người dùng để hiểu rõ nhu cầu, mong muốn và kỳ vọng của họ.

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

Lý do cần tổng hợp kết quả yêu cầu:

- Loại bỏ sự trùng lặp và mâu thuẫn
- Phân loại yêu cầu rõ ràng
- Tăng tính nhất quán và đầy đủ
- Tiết kiệm thời gian và công sức
- Cải thiện giao tiếp giữa các bên liên quan

3.Use case là gì?

Use case: Mô tả các chức năng mà hệ thống cung cấp cho người dùng.

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

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?

Include:Một use case bao gồm một use case khác
6.Quan hệ Extend giữa các use case là gì?
Extend:Một use case mở rộng chức năng cho một use case khác
7.Yêu cầu phi chức năng là gì?

Yêu cầu phi chức năng (Non-functional Requirement - NFR) là những yêu cầu không mô tả trực tiếp các chức năng mà hệ thống phải thực hiện, mà tập trung vào các khía cạnh vận hành, hiệu suất, bảo mật, khả năng sử dụng và bảo trì của hệ thống. Những yêu cầu này ảnh hưởng đến trải nghiệm người dùng và chất lượng hệ thống.

Các loại yêu cầu phi chức năng phổ biến:

- Hiệu suất (Performance): Độ trễ, tốc độ xử lý, thời gian phản hồi của hệ thống.
- Khả năng mở rộng (Scalability): Hệ thống có thể mở rộng để phục vụ nhiều người dùng hơn không?
- Bảo mật (Security): Các biện pháp đảm bảo dữ liệu không bị truy cập trái phép.
- Tính sử dụng (Usability): Độ thân thiện với người dùng, dễ học, dễ sử dụng.
- Khả năng bảo trì (Maintainability): Dễ dàng sửa lỗi, cập nhật hệ thống.
- Tính sẵn sàng (Availability): Hệ thống có thể hoạt động liên tục bao lâu mà không bị gián đoạn?
- Tính di động (Portability): Hệ thống có thể chạy trên nhiều nền tảng khác nhau không?

Ví dụ về yêu cầu phi chức năng:

- "Hệ thống phải xử lý tối thiểu 1000 yêu cầu/giây." (Hiệu suất)
- "Người dùng phải đăng nhập bằng xác thực hai yếu tố." (Bảo mật)
- "Hệ thống phải hoạt động liên tục 99.9% thời gian trong năm." (Tính sẵn sàng)

8.Mô hình nghiệp vụ là gì?

Mô hình nghiệp vụ (Business Model) là một bản mô tả chi tiết về cách tổ chức/doanh nghiệp hoạt động, cách tạo ra giá trị, quy trình kinh doanh, và các thành phần liên quan trong hệ thống. Nó giúp các bên liên quan hiểu rõ luồng công việc và xác định các yêu cầu cho hệ thống thông tin hỗ trợ.

Thành phần của mô hình nghiệp vụ:

1. Tác nhân (Actors): Các đối tượng tham gia vào hệ thống như khách hàng, nhân viên, nhà cung cấp.
2. Quy trình kinh doanh (Business Processes): Các bước thực hiện để cung cấp sản phẩm/dịch vụ.

3. Luồng công việc (Workflow): Cách dữ liệu và công việc được luân chuyển giữa các tác nhân.
4. Quy tắc nghiệp vụ (Business Rules): Các quy định, chính sách chi phối hoạt động của tổ chức.
5. Tài nguyên (Resources): Các nguồn lực như con người, công nghệ, tài chính.

Ví dụ về mô hình nghiệp vụ:

- Ngân hàng: Quy trình mở tài khoản, giao dịch, cấp tín dụng.
- Bệnh viện: Quy trình tiếp nhận bệnh nhân, khám chữa bệnh, thanh toán viện phí.
- Thương mại điện tử: Quy trình đặt hàng, thanh toán, vận chuyển.

Lợi ích của mô hình nghiệp vụ:

- Giúp hiểu rõ cách doanh nghiệp hoạt động.
- Xác định và tối ưu quy trình kinh doanh.
- Hỗ trợ việc xây dựng hệ thống thông tin phù hợp với thực tế.
- Cải thiện hiệu suất và chất lượng dịch vụ.

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

- **Điều kiện trước (Precondition)**: Là trạng thái hoặc điều kiện cần phải được thỏa mãn trước khi use case có thể bắt đầu. Nó mô tả bối cảnh hoặc yêu cầu ban đầu để use case được thực thi thành công. Ví dụ: "Người dùng đã đăng nhập vào hệ thống" là một điều kiện trước cho use case "Đặt hàng trực tuyến".
- **Điều kiện sau (Postcondition)**: Là trạng thái hoặc kết quả được đảm bảo sau khi use case hoàn thành thành công. Nó mô tả những gì hệ thống hoặc người dùng đạt được khi use case kết thúc. Ví dụ: "Hệ thống đã ghi nhận đơn hàng và gửi thông báo xác nhận cho người dùng" là một điều kiện sau cho use case "Đặt hàng trực tuyến".

Tóm lại:

- Điều kiện trước xác định "trạng thái bắt đầu".
- Điều kiện sau xác định "trạng thái kết thúc" của use case.

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

Khi trích các use case (xác định và mô tả các trường hợp sử dụng), cần chú ý các điểm sau để đảm bảo tính chính xác và hiệu quả:

1. **Hiểu rõ tác nhân (Actor)**: Xác định đúng ai hoặc cái gì (người dùng, hệ thống khác) sẽ tương tác với hệ thống trong use case. Ví dụ: "Khách hàng", "Quản trị viên", hay "Hệ thống thanh toán".
2. **Tập trung vào mục tiêu cụ thể**: Mỗi use case nên đại diện cho một mục tiêu rõ ràng mà tác nhân muốn đạt được (ví dụ: "Đăng ký tài khoản", "Thanh toán hóa đơn"), thay vì mô tả quá trình chi tiết.
3. **Sử dụng ngôn ngữ đơn giản, không kỹ thuật**: Viết use case sao cho dễ hiểu đối với cả nhóm phát triển và các bên liên quan không chuyên về kỹ thuật (stakeholders).
4. **Phân biệt giữa luồng chính và luồng phụ**:
    - Luồng chính (Main Flow): Mô tả kịch bản thành công cơ bản.
    - Luồng phụ (Alternative Flow): Bao gồm các trường hợp ngoại lệ hoặc lỗi (ví dụ: "Hệ thống báo lỗi nếu thông tin thẻ tín dụng không hợp lệ").
5. **Đảm bảo tính đầy đủ và nhất quán**: Các use case cần bao quát toàn bộ chức năng chính của hệ thống và không mâu thuẫn với nhau.
6. **Không đi quá chi tiết vào thiết kế**: Use case tập trung vào "cái gì" (hệ thống làm gì), không phải "làm thế nào" (cách hệ thống thực hiện).
7. **Xác định phạm vi rõ ràng**: Đảm bảo use case nằm trong ranh giới của hệ thống đang phân tích, tránh mở rộng quá mức sang các hệ thống khác.

Ví dụ lưu ý thực tế: Khi trích use case "Rút tiền từ ATM", cần ghi rõ điều kiện trước (như "Khách hàng đã chèn thẻ và nhập PIN đúng") và không mô tả chi tiết cách máy ATM đếm tiền (vì đó là thiết kế nội bộ).

CÂU HỎI TÌNH HUỐNG

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?

Giải thích thuật ngữ một cách đơn giản, dễ hiểu

- Tránh dùng từ chuyên môn quá phức tạp.
- Sử dụng các ví dụ thực tế, hình ảnh minh họa để giúp khách hàng dễ hình dung.
- Nếu cần, có thể tạo một danh sách thuật ngữ kèm giải thích trong tài liệu.

Thảo luận trực tiếp với khách hàng

- Tổ chức buổi họp hoặc trình bày để giải thích tài liệu.

- Lắng nghe phản hồi của khách hàng để điều chỉnh cách diễn đạt cho phù hợp.

Sử dụng ngôn ngữ gần gũi với khách hàng

- Thay vì dùng thuật ngữ chuyên môn, có thể diễn đạt theo cách khách hàng quen thuộc.
- Ví dụ, thay vì "API", có thể giải thích là "cách hệ thống giao tiếp với ứng dụng khác".

Cập nhật tài liệu để dễ hiểu hơn

- Viết lại các phần mô tả yêu cầu bằng cách sử dụng câu từ đơn giản hơn.
- Bổ sung chú thích hoặc sơ đồ minh họa nếu cần thiết.

Hướng dẫn khách hàng một cách chủ động

- Cung cấp tài liệu hướng dẫn ngắn gọn về các thuật ngữ quan trọng.
- Nếu khách hàng cần hiểu sâu hơn, có thể tổ chức buổi đào tạo ngắn.

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?

Xác định nguyên nhân thay đổi

- Hỏi rõ lý do khách hàng thay đổi yêu cầu (ví dụ: thay đổi nhu cầu kinh doanh, nhận phản hồi từ người dùng, v.v.).
- Xác định xem yêu cầu mới có thực sự cần thiết hay chỉ là ý tưởng chưa rõ ràng.

Thiết lập quy trình quản lý thay đổi yêu cầu

- Yêu cầu khách hàng cung cấp lý do rõ ràng cho mỗi thay đổi.
- Xây dựng quy trình phê duyệt thay đổi, bao gồm:
    - Đánh giá tác động của thay đổi đến tiến độ, chi phí, và các chức năng khác.
    - Thảo luận với khách hàng về ảnh hưởng của thay đổi trước khi thực hiện.

Định nghĩa phạm vi dự án ngay từ đầu

- Xác định rõ phạm vi công việc trong tài liệu yêu cầu (SRS - Software Requirement Specification).
- Nếu có thay đổi lớn, cần lập thêm phụ lục hợp đồng hoặc thỏa thuận bổ sung.

Sử dụng phương pháp phát triển linh hoạt (Agile)

- Nếu dự án cho phép, nên áp dụng mô hình Agile hoặc Scrum để dễ dàng thích nghi với thay đổi.
- Thực hiện phát triển theo từng giai đoạn nhỏ (iterative development) để khách hàng có thể kiểm tra và điều chỉnh sớm.

Thảo luận về chi phí và tiến độ

- Giải thích cho khách hàng rằng mỗi thay đổi có thể ảnh hưởng đến chi phí và thời gian hoàn thành.
- Nếu thay đổi quá nhiều, cần điều chỉnh lại ngân sách và thời gian.

Đảm bảo sự thống nhất trong yêu cầu

- Tổng hợp lại các yêu cầu mới và yêu cầu cũ để đảm bảo không có xung đột.
- Xác nhận lại với khách hàng danh sách yêu cầu cuối cùng trước khi triển khai.

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?

Tổ chức lại quá trình thu thập yêu cầu

- **Làm việc với khách hàng/stakeholder:** Đội phát triển cần tổ chức các buổi họp trực tiếp hoặc phỏng vấn với khách hàng (hoặc người dùng cuối) để làm rõ các yêu cầu. Hãy chuẩn bị sẵn các câu hỏi cụ thể như:
    - "Chức năng này được sử dụng trong tình huống nào?"
    - "Kết quả mong muốn cụ thể là gì?"
    - "Có ví dụ thực tế nào minh họa cho yêu cầu này không?"
- **Sử dụng kỹ thuật phân tích yêu cầu:** Áp dụng các phương pháp như User Stories, Use Case Diagram, hoặc bảng câu hỏi để biến những mô tả chung chung thành các yêu cầu cụ thể, có thể đo lường được (SMART: Specific, Measurable, Achievable, Relevant, Time-bound).

Tạo nguyên mẫu (Prototype)

- Nếu yêu cầu vẫn mơ hồ sau khi thảo luận, đội phát triển nên xây dựng một nguyên mẫu đơn giản (mockup hoặc giao diện cơ bản) dựa trên hiểu biết ban đầu.

- Đưa nguyên mẫu này cho khách hàng xem và lấy phản hồi. Điều này giúp cả hai bên hình dung rõ hơn về sản phẩm và điều chỉnh kỳ vọng.

Phân chia yêu cầu thành các phần nhỏ hơn

- Chia nhỏ các yêu cầu lớn, chung chung thành các nhiệm vụ cụ thể. Ví dụ:
    - Yêu cầu chung: "Hệ thống phải hỗ trợ quản lý bán hàng hiệu quả."
    - Yêu cầu cụ thể:
        - "Hệ thống cho phép nhập thông tin đơn hàng trong vòng 30 giây."
        - "Hệ thống tự động tính tổng doanh thu theo ngày/tháng."
        - "Hệ thống hiển thị báo cáo tồn kho theo thời gian thực."
- Sử dụng phương pháp Agile (như Scrum) để làm việc theo từng sprint, tập trung vào từng phần nhỏ và liên tục xác nhận với khách hàng.

Thiết lập tài liệu yêu cầu rõ ràng

- Soạn thảo một tài liệu **SRS (Software Requirement Specification)** hoặc ít nhất là một danh sách yêu cầu chi tiết, bao gồm:
    - Mục tiêu của chức năng.
    - Các điều kiện đầu vào/đầu ra.
    - Quy tắc nghiệp vụ (business rules).
- Yêu cầu khách hàng ký xác nhận tài liệu này để tránh hiểu lầm sau này.

Tăng cường giao tiếp nội bộ và với khách hàng

- **Trong đội phát triển:** Đảm bảo mọi thành viên (developer, tester, BA) đều hiểu rõ yêu cầu bằng cách tổ chức họp nội bộ thường xuyên.
- **Với khách hàng:** Thiết lập kênh liên lạc chính thức (email, công cụ quản lý dự án như Jira, Trello) để ghi nhận mọi thay đổi hoặc làm rõ yêu cầu.

Đánh giá lại quy trình quản lý dự án

- Nếu vấn đề xuất phát từ việc thiếu quy trình thu thập yêu cầu ban đầu, đội ngũ cần xem xét lại cách tiếp cận quản lý dự án. Có thể cần một Business Analyst (BA) hoặc Product Owner có kinh nghiệm để làm cầu nối giữa khách hàng và đội phát triển.

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?

Xác định phạm vi yêu cầu ban đầu

Hỏi khách hàng về mục tiêu chính của dự án.

Yêu cầu khách hàng mô tả vấn đề họ muốn giải quyết thay vì chỉ tập trung vào giải pháp cụ thể.

Xác định đối tượng sử dụng và bối cảnh áp dụng phần mềm.

Sử dụng kỹ thuật thu thập yêu cầu

Áp dụng một số phương pháp sau để làm rõ yêu cầu:

Phỏng vấn chi tiết: Hỏi khách hàng từng câu hỏi cụ thể để khai thác thông tin. Ví dụ:

Ai là người sử dụng chính?

Họ mong đợi tính năng gì?

Có hệ thống nào tương tự họ đã từng sử dụng không?

Workshop/brainstorming: Tổ chức buổi thảo luận với khách hàng để làm rõ các ý tưởng.

Quan sát quy trình làm việc hiện tại: Nếu phần mềm thay thế một hệ thống cũ, cần nghiên cứu cách hoạt động trước đó.

Use Case/User Story: Xây dựng tình huống sử dụng thực tế để khách hàng dễ hình dung hơn.

Tạo Prototype hoặc Wireframe

Nếu khách hàng không rõ ràng về yêu cầu, hãy xây dựng một prototype hoặc wireframe đơn giản để họ phản hồi.

Dùng Figma hoặc các công cụ vẽ sơ đồ UI/UX để mô phỏng phần mềm.

Lập tài liệu yêu cầu chi tiết

Sau khi thu thập được thông tin, viết tài liệu SRS (Software Requirements Specification) hoặc backlog để mô tả yêu cầu rõ ràng.

Gửi tài liệu này để khách hàng xác nhận trước khi phát triển.

Áp dụng mô hình phát triển linh hoạt (Agile)

Nếu yêu cầu quá mơ hồ, hãy chia nhỏ thành các Sprint ngắn và điều chỉnh liên tục dựa trên phản hồi khách hàng.

Thực hiện các buổi họp định kỳ để cập nhật và tinh chỉnh yêu cầu theo thực tế.

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?

Xác định mức độ trùng lặp

- Xem xét các Use Case có thực sự trùng lặp hoàn toàn hay chỉ có một phần chức năng giống nhau.
- Nếu có sự khác biệt nhỏ, hãy cân nhắc cách tổ chức lại để tránh dư thừa.

Sử dụng quan hệ Include

- Nếu nhiều Use Case có cùng một tập hợp hành động chung, tách phần chung thành một Use Case độc lập và sử dụng quan hệ «include» để tái sử dụng nó.
- Ví dụ:

Use Case A: "Đặt hàng"

Use Case B: "Thanh toán đơn hàng"

Nếu cả hai đều cần "Xác minh tài khoản khách hàng", có thể tạo Use Case C: "Xác minh tài khoản khách hàng" và dùng «include» trong A và B.

Xem xét quan hệ Generalization

- Nếu một số Use Case có chức năng tương tự nhưng có một số điểm khác biệt, bạn có thể dùng Generalization.
- Ví dụ:

Use Case: Thanh toán

Use Case con: Thanh toán bằng thẻ tín dụng

Use Case con: Thanh toán bằng PayPal

Cả hai đều kế thừa các bước chung của Use Case cha.

Sử dụng quan hệ Extend nếu phù hợp

· Nếu một Use Case có thể mở rộng một số chức năng nhất định tùy vào điều kiện, hãy sử dụng «extend».

· Ví dụ:

Use Case: Mua vé xem phim

Use Case mở rộng: Chọn ghế VIP (chỉ xảy ra nếu khách hàng chọn vé VIP).

Kiểm tra lại với khách hàng

Trước khi chỉnh sửa mô hình, cần xác nhận lại với khách hàng hoặc nhóm phân tích nghiệp vụ để đảm bảo cấu trúc mới phản ánh đúng yêu cầu thực tế.

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?

Xác định rõ ràng sự khác biệt về ý nghĩa

- Tổ chức cuộc họp giữa đội phát triển và khách hàng để làm rõ những từ khóa đang gây hiểu nhầm.
- Đặt câu hỏi để hiểu quan điểm của cả hai bên và xác định chính xác từ ngữ đang gây tranh cãi.

Giải thích và minh họa rõ ràng

- Giải thích từ khóa chuyên môn bằng ngôn ngữ đơn giản và dễ hiểu hơn, tránh sử dụng thuật ngữ quá kỹ thuật.
- Cung cấp ví dụ minh họa cụ thể để khách hàng dễ hình dung, nếu cần có thể sử dụng hình ảnh hoặc sơ đồ.

Đảm bảo sự đồng thuận về định nghĩa

- Cùng khách hàng thống nhất một định nghĩa chính thức cho các từ khóa chuyên môn, đảm bảo cả đội phát triển và khách hàng đều hiểu giống nhau.
- Cập nhật tài liệu yêu cầu để phản ánh đúng định nghĩa đã thống nhất.

Tạo một bảng thuật ngữ

- Biên soạn danh sách thuật ngữ và định nghĩa chính xác các từ khóa chuyên môn trong tài liệu dự án.
- Cung cấp tài liệu này cho khách hàng và đội phát triển để tránh hiểu nhầm trong suốt quá trình phát triển.

Giao tiếp liên tục

- Duy trì giao tiếp thường xuyên giữa đội phát triển và khách hàng để tránh các sự cố tương tự trong tương lai.
- Khuyến khích khách hàng đặt câu hỏi ngay khi không hiểu rõ về bất kỳ thuật ngữ nào.

Cập nhật tài liệu và hợp đồng

- Nếu cần, cập nhật lại hợp đồng hoặc tài liệu yêu cầu để đảm bảo rằng các từ khóa và yêu cầu đều được hiểu thống nhất, tránh hiểu nhầm trong quá trình phát triển.

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.

Khi đội phát triển nhận ra rằng một số chức năng quan trọng bị bỏ sót sau khi hoàn thành việc trích xuất **Use Case**, họ có thể thực hiện các bước sau để khắc phục:

Xác định nguyên nhân bỏ sót:

- Do thu thập yêu cầu chưa đầy đủ?
- Do hiểu sai nghiệp vụ?
- Do khách hàng chưa cung cấp đủ thông tin?

Trao đổi lại với khách hàng và các bên liên quan:

- Tổ chức các cuộc họp để xác nhận lại các chức năng bị thiếu.
- Sử dụng **câu hỏi, phỏng vấn, quan sát quy trình thực tế** để đảm bảo không còn sót yêu cầu nào khác.

Bổ sung và cập nhật mô hình Use Case:

- Thêm mới các **Use Case** bị thiếu.
- Nếu cần, cập nhật các mối quan hệ **Include, Extend** giữa các Use Case.
- Kiểm tra xem việc bổ sung có ảnh hưởng đến các Use Case hiện tại không.

Đánh giá tác động đến thiết kế và phát triển:

- Kiểm tra xem việc bổ sung có ảnh hưởng đến kiến trúc, cơ sở dữ liệu, hoặc mã nguồn đã triển khai không.

- Nếu có, lập kế hoạch điều chỉnh phù hợp.

Cập nhật lại tài liệu:

- Điều chỉnh tài liệu đặc tả Use Case.
- Cập nhật sơ đồ Use Case nếu cần.

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 muốn bổ sung chức năng mới trong quá trình xây dựng **mô hình nghiệp vụ**, đội phát triển có thể thực hiện các bước sau:

Phân tích yêu cầu mới:

- Xác định xem yêu cầu mới có phù hợp với mục tiêu hệ thống không.
- Kiểm tra xem nó có xung đột với các chức năng hiện có không.

Trao đổi và xác nhận lại với khách hàng:

- Giải thích tác động của việc bổ sung chức năng mới.
- Nếu có thể, đề xuất giải pháp thay thế nếu yêu cầu không khả thi.

Đánh giá tác động lên hệ thống:

- Xem xét ảnh hưởng đến mô hình nghiệp vụ, quy trình kinh doanh.
- Kiểm tra sự ảnh hưởng đến chi phí, thời gian, tài nguyên.

Cập nhật mô hình nghiệp vụ:

- Thêm mới quy trình kinh doanh, tác nhân, luồng công việc nếu cần.
- Điều chỉnh quy tắc nghiệp vụ và mô tả chức năng.

Cập nhật tài liệu và kế hoạch phát triển:

- Sửa đổi tài liệu đặc tả yêu cầu nghiệp vụ (SRS).
- Điều chỉnh kế hoạch phát triển để tích hợp chức năng mới.

Thông báo và điều chỉnh tiến độ dự án:

- Nếu việc bổ sung chức năng ảnh hưởng đến tiến độ, cần thảo luận với khách hàng để điều chỉnh kế hoạch.
- Xác định xem có cần thêm nhân lực hoặc ngân sách không.

Thực hiện và kiểm thử:

- Phát triển chức năng mới theo mô hình cập nhật.
- Kiểm thử để đảm bảo tính tương thích và đúng yêu cầu.

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.

Khi nhóm phát triển không hiểu rõ quy trình nghiệp vụ (business process), việc mô tả use case sẽ thiếu chính xác hoặc không đầy đủ. Dưới đây là các giải pháp để khắc phục:

1. **Tổ chức buổi làm việc với các bên liên quan (Stakeholders)**:
   - Mời khách hàng, người dùng cuối hoặc chuyên gia nghiệp vụ tham gia để giải thích quy trình hiện tại.
   - Sử dụng các câu hỏi như: "Quy trình này diễn ra như thế nào?", "Ai tham gia?", "Mục tiêu cuối cùng là gì?" để làm rõ.
2. **Thực hiện phân tích quy trình nghiệp vụ (Business Process Analysis)**:
   - Vẽ sơ đồ luồng công việc (workflow diagram) hoặc biểu đồ hoạt động (activity diagram) để hình dung cách quy trình vận hành.
   - Xác định các bước chính, đầu vào (input), đầu ra (output) và các tác nhân liên quan.
3. **Quan sát thực tế (Observation)**:
   - Nếu có thể, nhóm nên đến (tại chỗ) để quan sát cách người dùng thực hiện quy trình trong môi trường thực tế (ví dụ: quan sát nhân viên ngân hàng xử lý giao dịch).
4. **Sử dụng tài liệu hiện có**:
   - Xem xét tài liệu quy trình, hướng dẫn sử dụng hoặc báo cáo nội bộ của khách hàng để hiểu rõ hơn về nghiệp vụ.
5. **Tạo bản nháp use case sơ bộ và lấy phản hồi**:
   - Viết một phiên bản use case đơn giản dựa trên hiểu biết ban đầu, sau đó gửi cho khách hàng hoặc chuyên gia nghiệp vụ để chỉnh sửa và bổ sung.
6. **Đào tạo nhóm về lĩnh vực nghiệp vụ**:
   - Nếu dự án thuộc một lĩnh vực đặc thù (như y tế, tài chính), tổ chức buổi đào tạo ngắn hạn để nhóm hiểu các khái niệm cơ bản.

Đề xuất cụ thể: Bắt đầu bằng cách tổ chức một buổi họp với khách hàng để vẽ sơ đồ quy trình nghiệp vụ, sau đó dùng thông tin này để viết các use case cơ bản và tinh chỉnh dần qua các vòng phản hồi.

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?

Việc khách hàng yêu cầu thêm chức năng mới trong giai đoạn thiết kế chi tiết là tình huống phổ biến trong phát triển phần mềm. Đội phát triển cần xử lý một cách có hệ thống để cân bằng giữa yêu cầu mới và tiến độ dự án. Dưới đây là các bước xử lý:

1. **Đánh giá yêu cầu mới**:
    - Ghi nhận yêu cầu từ khách hàng (ví dụ: qua email, biên bản họp).
    - Phân tích yêu cầu: Chức năng này là gì? Nó ảnh hưởng đến những phần nào của hệ thống (giao diện, cơ sở dữ liệu, logic xử lý)?
2. **Xem xét tác động (Impact Analysis)**:
    - Kiểm tra mức độ ảnh hưởng đến thiết kế hiện tại: Có cần thay đổi kiến trúc không? Có ảnh hưởng đến các chức năng khác không?
    - Ước lượng thời gian, công sức và chi phí để triển khai chức năng mới.
3. **Thương thảo với khách hàng**:
    - Trình bày tác động của yêu cầu mới lên tiến độ và ngân sách hiện tại.
    - Đề xuất các lựa chọn:
        - **Thêm vào ngay**: Điều chỉnh kế hoạch và tài nguyên để tích hợp chức năng mới.
        - **Hoãn lại**: Đưa chức năng vào phiên bản tiếp theo (nếu dự án theo mô hình lặp - iterative).
        - **Từ chối (nếu cần)**: Giải thích nếu yêu cầu không khả thi trong phạm vi hiện tại.
4. **Cập nhật tài liệu dự án**:
    - Nếu đồng ý thêm chức năng, cập nhật tài liệu yêu cầu (requirements specification), use case, thiết kế hệ thống và kế hoạch dự án.
    - Đảm bảo tất cả thành viên trong nhóm được thông báo về thay đổi.
5. **Điều chỉnh tiến độ và tài nguyên**:
    - Nếu cần, phân bổ thêm thời gian hoặc nhân lực để hoàn thành yêu cầu mới mà không ảnh hưởng chất lượng.
6. **Áp dụng quy trình quản lý thay đổi (Change Management)**:
    - Nếu dự án có quy trình chính thức (như trong mô hình Waterfall), yêu cầu khách hàng ký duyệt thay đổi (Change Request).
    - Trong mô hình Agile, đưa yêu cầu vào backlog và ưu tiên trong sprint tiếp theo.

Đề xuất cụ thể: Đội phát triển nên họp nội bộ để đánh giá tác động trong 1-2 ngày, sau đó tổ chức họp với khách hàng để thống nhất giải pháp. Nếu chức năng không quá phức tạp, có thể tích hợp ngay; nếu không, đề xuất hoãn lại để đảm bảo tiến độ.