CHƯƠNG 2

Trắc nghiệm

Câu hỏi 1: Workflow nào trong tiến trình phát triển phần mềm chịu trách nhiệm thu thập yêu cầu từ khách hàng?

Đáp án: **B. Workflow lấy yêu cầu**

Câu hỏi 2: Pha nào trong tiến trình thống nhất (Unified Process) tập trung vào

việc phân tích rủi ro và xây dựng kiến trúc ban đầu?

Đáp án: **B. Pha làm rõ**

Câu hỏi 3: Mô hình CMM mức nào yêu cầu quy trình phát triển phần mềm phải

được quản lý định lượng?

Đáp án: **C. Mức 4**

Câu hỏi 4: Các pha trong tiến trình thống nhất bao gồm:

Đáp án: **B. Khởi đầu, làm rõ, xây dựng, chuyển giao**

Câu hỏi 5: Trong tiến trình thống nhất, workflow nào thực hiện sau cùng?

Đáp án: **D. Workflow kiểm thử**

Câu hỏi 6: Mô hình CMM mức 1 có đặc điểm gì?

Đáp án: **C. Quy trình không ổn định, phụ thuộc vào cá nhân**

Câu hỏi 7: Tiến trình thống nhất là một ví dụ của mô hình nào?

Đáp án: **B. Mô hình lặp và tăng trưởng**

Câu hỏi 8: Trong mô hình CMM mức 5, quy trình phát triển phần mềm

có đặc điểm gì?

Đáp án: **A. Quy trình được cải tiến liên tục**

Câu hỏi 9: Workflow thiết kế bao gồm việc thực hiện hoạt động nào?

Đáp án: **C. Thiết kế kiến trúc và chi tiết hệ thống**

Câu hỏi 10: CMM viết tắt của cụm từ nào?

Đáp án: **B. Capability Maturity Model**

Câu hỏi ngắn

1. **Pha khởi đầu trong tiến trình thống nhất là gì?**

Pha khởi đầu (Inception):

Mục tiêu chính là hiểu rõ các yêu cầu cơ bản và đánh giá tính khả thi của dự án.

Pha khởi đầu trong tiến trình thống nhất giúp nhóm phát triển phần mềm có cái nhìn rõ ràng về mục tiêu, yêu cầu và tính khả thi của dự án. Pha này cũng giúp xác định các yếu tố rủi ro và lập kế hoạch cho các giai đoạn tiếp theo, như thiết kế, triển khai và kiểm thử.

2. **Mục tiêu của workflow lấy yêu cầu là gì?**

Mục tiêu: Xác định và ghi nhận tất cả các yêu cầu từ phía khách hàng.

Kết quả: Tài liệu yêu cầu phần mềm được duyệt bởi khách hàng.

3. **Tiến trình thống nhất gồm bao nhiêu pha chính?**

4 pha: Pha khởi đầu (Inception), Pha làm rõ (Elaboration), Pha xây dựng (Construction), Pha chuyển giao (Transition)

4. **Sự khác nhau giữa CMM mức 2 và mức 3 là gì?**

CMM mức 2 – Managed (Quản lý):

Quy trình được quản lý ở mức cơ bản.

Các hoạt động như lập kế hoạch, quản lý rủi ro được thực hiện.

CMM mức 3 – Defined (Định nghĩa):

Quy trình được định nghĩa rõ ràng và nhất quán trong toàn tổ chức.

Các tiêu chuẩn quy trình được xây dựng và áp dụng.

5. **Workflow kiểm thử có nhiệm vụ gì?**

Mục tiêu: Đảm bảo rằng phần mềm hoạt động đúng như mong đợi.

Kết quả: Phần mềm đạt tiêu chuẩn chất lượng và sẵn sàng triển khai.

6. **Mô hình CMM có bao nhiêu mức?**

5 mức: 1 – Initial (Ban đầu), 2 – Managed (Quản lý), 3 – Defined (Định nghĩa), 4 – Quantitatively Managed (Quản lý định lượng), 5 – Optimizing (Tối ưu hóa)

7. **Khác biệt giữa mô hình thác nước và mô hình lặp là gì?**

Mô hình thác nước (Waterfall Model):

Đặc điểm: Phát triển tuyến tính, từng pha hoàn thành trước khi chuyển sang pha tiếp theo.

Ưu điểm: Dễ quản lý và theo dõi tiến độ.

Nhược điểm: Khó thích nghi khi yêu cầu thay đổi.

Mô hình lặp và tăng trưởng (Iterative and Incremental Model):

Đặc điểm: Phát triển phần mềm theo từng đợt lặp lại, mỗi đợt cải tiến dần phần mềm.

Ưu điểm: Giảm thiểu rủi ro và thích nghi tốt với thay đổi yêu cầu.

Nhược điểm: Tốn nhiều công sức quản lý và kiểm thử.

8. **Tiến trình thống nhất có phải là mô hình lặp không?**

Có, tiến trình thống nhất (Unified Process) là một ví dụ của mô hình lặp và tăng trưởng.

9. **Mục đích của workflow thiết kế là gì?**

Mục tiêu: Thiết kế chi tiết các thành phần phần mềm dựa trên kết quả phân tích.

Kết quả: Tài liệu thiết kế chi tiết và sơ đồ UML.

10. **CMM mức 5 tập trung vào điều gì?**

Mục tiêu là đạt được sự hoàn hảo trong phát triển phần mềm.

Câu hỏi thảo luận nhóm

Câu 1: Thảo luận về vai trò của từng workflow trong tiến trình phát triển phần mềm.

Trong quy trình phát triển phần mềm, mỗi workflow đóng vai trò quan trọng nhằm đảm bảo hệ thống được xây dựng một cách hiệu quả, đáp ứng đúng yêu cầu và đạt chất lượng cao. Theo nhóm em, vai trò cụ thể của từng workflow là:

1. Workflow lấy yêu cầu: Đây là giai đoạn nền tảng, giúp xác định rõ nhu cầu của khách hàng hoặc người dùng cuối. Việc thu thập yêu cầu đầy đủ và chính xác sẽ đảm bảo phần mềm được phát triển theo đúng mục

2. Workflow phân tích: Sau khi xác định yêu cầu, bước phân tích giúp làm rõ các chức năng của hệ thống, xác định các ràng buộc kỹ thuật và đánh giá tính khả thi của dự án. Workflow này đóng vai trò quan trọng trong việc nhận diện rủi ro và đề xuất giải pháp tối ưu, tạo tiền đề cho thiết kế và triển khai hiệu quả.
3. Workflow thiết kế: Giai đoạn thiết kế xác định kiến trúc tổng thể của hệ thống, bao gồm thiết kế giao diện, mô hình dữ liệu, cấu trúc xử lý và các thành phần kỹ thuật liên quan. Một thiết kế tốt không chỉ đảm bảo hiệu suất và tính ổn định của phần mềm mà còn giúp hệ thống dễ bảo trì và mở rộng trong tương lai.
4. Workflow cài đặt: Đây là giai đoạn hiện thực hóa thiết kế thông qua việc lập trình. Các lập trình viên phát triển phần mềm theo đúng tiêu chuẩn kỹ thuật, đảm bảo mã nguồn tối ưu, dễ hiểu và có khả năng bảo trì.
5. Workflow kiểm thử: Trước khi đưa vào vận hành, phần mềm cần trải qua quá trình kiểm thử nghiêm ngặt nhằm phát hiện và khắc phục lỗi. Kiểm thử bao gồm nhiều cấp độ như kiểm thử chức năng, kiểm thử hiệu năng, kiểm thử bảo mật và kiểm thử khả năng sử dụng. Workflow này đảm bảo hệ thống hoạt động ổn định, đáp ứng yêu cầu kỹ thuật và mang lại trải nghiệm tốt cho người dùng.

Mỗi workflow trong quy trình phát triển phần mềm đều có vai trò không thể thiếu, giúp tối ưu hóa quy trình làm việc, nâng cao chất lượng sản phẩm và đảm bảo dự án được triển khai thành công.

Câu 2: Phân biệt mô hình vòng đời thác nước và tiến trình thống nhất.

1. Mô hình Vòng đời Thác nước (Waterfall Model): Mô hình Vòng đời Thác nước là phương pháp phát triển phần mềm tuyến tính, trong đó mỗi giai đoạn phải hoàn thành trước khi chuyển sang giai đoạn tiếp theo.

- Đặc điểm: Quy trình phát triển theo thứ tự chặt chẽ từ phân tích, thiết kế, cài đặt, kiểm thử đến bảo trì, không có sự quay lại hoặc lặp lại giữa các pha, phù hợp với các dự án có yêu cầu rõ ràng, ít thay đổi.

- Ưu điểm: Dễ quản lý do có quy trình cụ thể và tài liệu chi tiết, phù hợp với dự án nhỏ, yêu cầu rõ ràng ngay từ đầu.

- Nhược điểm: Thiếu linh hoạt, khó thay đổi khi yêu cầu khách hàng thay đổi, phát hiện lỗi muộn, chi phí sửa đổi cao.

2. Tiến trình Thống nhất (Unified Process - UP): Tiến trình Thống nhất là mô hình phát triển phần mềm lặp và gia tăng, tập trung vào việc giải quyết rủi ro sớm trong dự án.

- Các pha trong Tiến trình Thống nhất:

    + Pha Khởi đầu (Inception): Xác định yêu cầu cốt lõi, đánh giá tính khả thi.

    + Pha Làm rõ (Elaboration): Phân tích và thiết kế hệ thống, giải quyết rủi ro.

    + Pha Xây dựng (Construction): Lập trình, kiểm thử và phát triển phần mềm.

    + Pha Chuyển giao (Transition): Bàn giao, triển khai phần mềm cho khách hàng.

- Đặc điểm: Cho phép lặp lại các pha, có thể điều chỉnh dựa trên phản hồi, xác định và giải quyết rủi ro từ sớm, phù hợp với dự án phức tạp, yêu cầu thay đổi linh hoạt.
- Ưu điểm: Giảm thiểu rủi ro nhờ phát hiện sớm các vấn đề, tích hợp kiểm thử sớm, giúp phát hiện lỗi kịp thời, phù hợp với dự án có yêu cầu thay đổi liên tục.
- Nhược điểm: Yêu cầu tài nguyên và kỹ năng quản lý cao, quá trình phát triển phức tạp, khó thực hiện đối với nhóm nhỏ.

Câu 3: Thảo luận về các ưu và nhược điểm của mô hình lặp và tăng trưởng.

Mô hình lặp là một phương pháp tiếp cận mạnh mẽ trong phát triển phần mềm, nhấn mạnh vào cải tiến liên tục và khả năng thích ứng. Mô hình này cho phép các nhóm phát triển phần mềm theo từng bước nhỏ, dễ quản lý, có thể mang lại một số lợi thế:

- Tính linh hoạt được nâng cao: Một trong những lợi ích chính của mô hình lặp là tính linh hoạt của nó. Khi các yêu cầu phát triển, các nhóm có thể điều chỉnh kế hoạch và ưu tiên của mình dựa trên phản hồi từ các bên liên quan. Khả năng thích ứng này rất quan trọng trong môi trường phát triển nhanh như hiện nay, nơi nhu cầu của người dùng có thể thay đổi nhanh chóng.
- Phát hiện sớm các vấn đề: Bằng cách chia nhỏ dự án thành các lần lặp, các nhóm có thể xác định và giải quyết các vấn đề ngay từ đầu trong quá trình phát triển. Mỗi lần lặp bao gồm thử nghiệm và phản hồi, giúp phát hiện lỗi trước khi chúng trở thành vấn đề lớn hơn. Cách tiếp cận chủ động này có thể tiết kiệm thời gian và tài nguyên trong thời gian dài.
- Cải thiện sự tham gia của người dùng: Mô hình lặp lại khuyến khích tương tác thường xuyên với người dùng, cho phép họ cung cấp phản hồi về mỗi lần gia tăng. Sự tham gia này không chỉ giúp tinh chỉnh sản phẩm mà còn đảm bảo rằng đầu ra cuối cùng phù hợp chặt chẽ với kỳ vọng của người dùng. Người dùng cảm thấy tham gia nhiều hơn vào quá trình phát triển, điều này có thể dẫn đến sự hài lòng cao hơn với sản phẩm cuối cùng.
- Chuyển giao từng phần: Với mô hình lặp lại, các nhóm có thể cung cấp các thành phần chức năng của phần mềm vào cuối mỗi lần lặp lại. Việc cung cấp gia tăng này

cho phép người dùng bắt đầu sử dụng các phần của hệ thống sớm hơn, cung cấp giá trị ngay từ đầu vòng đời dự án. Nó cũng cho phép các nhóm thu thập phản hồi của người dùng về các thành phần này, có thể cung cấp thông tin cho các lần lặp lại trong tương lai.

- Quản lý rủi ro: Phương pháp lặp lại giúp quản lý rủi ro hiệu quả. Bằng cách đánh giá dự án vào cuối mỗi lần lặp lại, các nhóm có thể đánh giá lại các chiến lược của mình và thực hiện các điều chỉnh cần thiết. Đánh giá liên tục này giúp giảm thiểu rủi ro liên quan đến các yêu cầu thay đổi hoặc các thách thức không lường trước được.
- Cải tiến liên tục: Mỗi lần lặp lại đóng vai trò là một cơ hội học tập. Các nhóm có thể phản ánh về quy trình và kết quả của mình, xác định các lĩnh vực cần cải thiện. Văn hóa cải tiến liên tục này thúc đẩy sự đổi mới và nâng cao chất lượng chung của phần mềm đang được phát triển.

Tuy nhiên cũng tồn tại một số nhược điểm của mô hình lặp và gia tăng:

- Phạm vi mở rộng : Tính linh hoạt của mô hình lặp có thể dẫn đến phạm vi mở rộng nếu những thay đổi không được quản lý hiệu quả.
- Tốn nhiều tài nguyên : Việc lặp lại thường xuyên có thể đòi hỏi nhiều tài nguyên và thời gian hơn, đặc biệt là nếu liên quan đến việc thử nghiệm mở rộng

Dù có một số hạn chế, mô hình lặp vẫn là một phương pháp phổ biến trong phát triển phần mềm hiện đại, giúp đảm bảo chất lượng và tối ưu trải nghiệm người dùng.

Câu 4: Vì sao mô hình CMM được sử dụng rộng rãi trong quản lý chất lượng phần mềm?

- CMM (Capability Maturity Model) là một mô hình đánh giá và trưởng thành năng lực, được phát triển bởi Viện Kỹ nghệ Phần mềm (SEI) của Đại học Carnegie Mellon.
- Mô hình này cung cấp một khung tham chiếu để đánh giá mức độ trưởng thành của quy trình phát triển phần mềm của một tổ chức.
- CMM giúp các tổ chức xác định các điểm mạnh và điểm yếu trong quy trình của mình, từ đó đưa ra các biện pháp cải tiến phù hợp.
- Việc áp dụng CMM giúp các tổ chức nâng cao chất lượng phần mềm, giảm thiểu rủi ro và tăng cường hiệu quả hoạt động.
- CMM cũng giúp các tổ chức tạo ra một môi trường làm việc chuyên nghiệp và có tổ chức hơn.

Câu 5: Thảo luận về các khó khăn khi áp dụng mô hình CMM trong thực tế.

- **Tốn kém chi phí và thời gian:** Việc đánh giá và triển khai CMM đòi hỏi đầu tư đáng kể về thời gian và nguồn lực.
- **Yêu cầu sự thay đổi văn hóa:** CMM yêu cầu các tổ chức thay đổi cách làm việc truyền thống và áp dụng các quy trình mới, điều này có thể gặp phải sự kháng cự từ nhân viên.
- **Khó khăn trong việc đo lường:** Việc đo lường hiệu quả của các cải tiến theo CMM có thể gặp khó khăn, đặc biệt là trong các tổ chức nhỏ.
- **Cần sự cam kết từ lãnh đạo:** Việc áp dụng CMM thành công đòi hỏi sự cam kết và hỗ trợ mạnh mẽ từ lãnh đạo cấp cao.
- **Không phù hợp với mọi tổ chức:** CMM được thiết kế cho các tổ chức lớn và có quy trình phức tạp, có thể không phù hợp với các tổ chức nhỏ hoặc các dự án đơn giản.

Câu 6: Đề xuất các giải pháp để cải tiến quy trình phát triển phần mềm.

- **Áp dụng phương pháp Agile:** Agile là một phương pháp phát triển phần mềm linh hoạt, giúp tăng cường sự hợp tác, phản hồi nhanh và khả năng thích ứng với thay đổi.
- **Tự động hóa quy trình:** Tự động hóa các tác vụ lặp đi lặp lại giúp giảm thiểu sai sót và tăng hiệu quả.
- **Tăng cường giao tiếp và hợp tác:** Tạo môi trường làm việc cởi mở, khuyến khích giao tiếp và chia sẻ thông tin giữa các thành viên trong nhóm.
- **Đào tạo và phát triển nhân viên:** Đầu tư vào đào tạo và phát triển kỹ năng cho nhân viên để họ có thể áp dụng các quy trình và công nghệ mới.
- **Sử dụng công cụ hỗ trợ:** Sử dụng các công cụ quản lý dự án, kiểm soát phiên bản và kiểm thử tự động để nâng cao hiệu quả.
- **Đo lường và đánh giá thường xuyên:** Thiết lập các chỉ số đo lường hiệu quả và thực hiện đánh giá thường xuyên để xác định các lĩnh vực cần cải tiến.
- **Học hỏi từ kinh nghiệm:** Tổ chức các buổi họp rút kinh nghiệm sau mỗi dự án để chia sẻ bài học và cải tiến quy trình.

7 Phân tích ưu điểm của việc áp dụng tiến trình thông nhất trong các dự án lớn.

Việc áp dụng tiến trình thông nhất (Unified Process) trong các dự án lớn mang lại nhiều ưu điểm, như:

- **Giảm rủi ro:** Phân chia công việc thành các giai đoạn nhỏ, cho phép phát hiện và xử lý vấn đề sớm.
- **Hỗ trợ làm việc nhóm:** Tăng cường phối hợp giữa các thành viên nhờ quy trình chuẩn hóa.
- **Tái sử dụng:** Khuyến khích sử dụng các thành phần đã có sẵn, tiết kiệm thời gian và chi phí.
- **Tập trung vào yêu cầu:** Đảm bảo hiểu rõ và đáp ứng tốt nhu cầu của khách hàng thông qua các vòng lặp phát triển.
- **Tính cấu trúc rõ ràng:** Tiến trình được chia thành các pha (khởi tạo, chuẩn bị, xây dựng, triển khai), giúp quản lý dự án dễ dàng hơn.

8 Thảo luận về sự cần thiết của việc kiểm thử trong tiến trình chuẩn bị thông nhất.
Việc kiểm thử trong pha chuẩn bị của tiến trình thông nhất rất cần thiết vì:

- **Đảm bảo chất lượng sớm:** Phát hiện lỗi hoặc vấn đề ngay từ đầu, tránh chi phí sửa chữa lớn ở các pha sau.
- **Lập kế hoạch hiệu quả:** Kiểm thử giúp xác định các rủi ro tiềm ẩn, từ đó xây dựng kế hoạch kiểm thử chi tiết cho các pha sau.
- **Tăng độ tin cậy:** Giúp đội phát triển tự tin hơn khi chuyển sang pha xây dựng với cơ sở vững chắc.
- **Xác minh yêu cầu:** Đảm bảo rằng các yêu cầu ban đầu được hiểu đúng và có thể triển khai được.

9 So sánh mô hình CMMI mức 4 và mức 5

- **CMMI mức 4 (Quản lý định lượng):**
  - Quản lý hiệu suất dự án thông qua các chỉ số cụ thể, nhưng vẫn có thể có biến động nhỏ.
  - Tập trung vào đo lường và kiểm soát quy trình bằng dữ liệu định lượng.
  - Yêu cầu tổ chức có khả năng thu thập và phân tích dữ liệu để cải tiến liên tục.
- **CMMI mức 5 (Tối ưu hóa):**
  - Cao hơn mức 4, tập trung vào tối ưu hóa quy trình thông qua cải tiến liên tục dựa trên dữ liệu.
  - Có khả năng dự đoán và ngăn ngừa vấn đề trước khi xảy ra nhờ phân tích sâu hơn.
  - Yêu cầu tổ chức đã đạt được tính ổn định cao và áp dụng các phương pháp tiên tiến.

    Sự khác biệt chính nằm ở mức độ tự động hóa, dự đoán và tối ưu hóa quy trình, với mức 5 vượt trội hơn về tính tiên tiến và hiệu quả.

10 Đề xuất cách tối ưu hoạt động nhóm trong workflow lập yêu cầu.

 Rõ ràng hóa vai trò: Phân công nhiệm vụ cụ thể cho từng thành viên, đảm bảo ai cũng hiểu trách nhiệm của mình.
 Sử dụng công cụ quản lý: Áp dụng các phần mềm như Jira, Trello hoặc Confluence để theo dõi tiến độ và giao tiếp hiệu quả.
 Tổ chức họp định kỳ: Thực hiện các buổi họp ngắn (stand-up meeting) để cập nhật tình hình, giải quyết vấn đề nhanh chóng.
 Đào tạo và giao tiếp: Đảm bảo tất cả thành viên hiểu rõ quy trình và yêu cầu, giảm thiểu hiểu lầm.
 Phân tích phản hồi: Thu thập ý kiến từ khách hàng và đội ngũ để cải tiến cách làm việc, tăng hiệu suất.

Câu hỏi tình huống

1. **Mỗi công ty phát triển phần mềm đều khó khăn khi yêu cầu của khách hàng liên tục thay đổi trong pha xây dựng. Dối phát triển nên làm gì để giải quyết vấn đề này?**
   Để giải quyết vấn đề này, đội phát triển nên áp dụng phương pháp linh hoạt như Agile, tập trung vào việc giao tiếp thường xuyên với khách hàng, sử dụng các cuộc họp định kỳ (như sprint review) để cập nhật yêu cầu, và duy trì tính linh hoạt trong quy trình phát triển. Điều này giúp điều chỉnh nhanh chóng khi có thay đổi.
2. **Trong pha chuẩn bị của tiến trình thông nhất, khách hàng yêu cầu bổ sung thêm tính năng mới. Đội phát triển nên xử lý ra sao?**
   Đội phát triển nên đánh giá tác động của tính năng mới đến thời gian, chi phí và phạm vi dự án. Sau đó, thảo luận với khách hàng để ưu tiên tính năng, có thể điều chỉnh kế hoạch hoặc đưa ra lộ trình bổ sung trong các giai đoạn sau nếu cần.
3. **Dự án phát triển phần mềm bị trì hoãn do lỗi phát sinh liên tục trong quá trình kiểm thử. Làm trường dự án, bạn sẽ làm gì?**
   Là trưởng dự án, bạn nên phân tích nguyên nhân gốc rễ của các lỗi, cải thiện quy trình kiểm thử (như tăng cường kiểm thử tự động hoặc kiểm thử sớm hơn), phân bổ lại nguồn lực nếu cần, và phối hợp chặt chẽ với đội phát triển để giảm thiểu lỗi trong tương lai.
4. **Trong workflow thiệt kế, kiến trúc sư phần mềm muốn thay đổi thiết kế ban đầu để cải thiện hiệu suất. Đội phát triển nên xử lý thế nào?**
   Đội phát triển nên đánh giá lợi ích và rủi ro của thay đổi thiết kế, tham khảo ý kiến các bên liên quan (bao gồm khách hàng nếu cần), và nếu thay đổi hợp lý, lập kế hoạch cập nhật mà không làm gián đoạn tiến độ tổng thể.
5. **Khách hàng yêu cầu rút ngắn thời gian phát triển dự án mà không thay đổi yêu cầu. Đội phát triển nên phân ứng ra sao?**
   Đội phát triển cần làm rõ tính khả thi, có thể đề xuất tăng cường nguồn lực, áp dụng các công cụ tự động hóa, hoặc ưu tiên các tính năng quan trọng nhất. Tuy nhiên, cần cảnh báo khách hàng về rủi ro chất lượng nếu thời gian bị rút ngắn quá mức.
6. **Mỗi công ty muốn áp dụng mô hình CMMI nhưng gặp khó khăn do thiếu nguồn lực. Hãy đề xuất giải pháp.**

Giải pháp có thể bao gồm bắt đầu từ cấp độ CMMI cơ bản (Level 1 hoặc 2), đào tạo nhân sự, sử dụng tư vấn chuyên nghiệp, và triển khai dần dần theo khả năng tài chính và nhân lực của công ty.

7. **Trong workflow lập yêu cầu, khách hàng liên tục cập thông tin không rõ ràng. Đội phát triển có vấn đề gì?**
   Vấn đề chính là sự thiếu nhất quán và hiểu biết rõ ràng về yêu cầu, dẫn đến khó khăn trong thiết kế và phát triển. Đội cần tổ chức các buổi làm việc trực tiếp hoặc sử dụng tài liệu chi tiết hơn để xác nhận yêu cầu với khách hàng.
8. **Mất dự án gấp rút ra cao trong pha kiểm tra do thiếu tài liệu yêu cầu rõ ràng. Đội phát triển nên làm gì?**
   Đội nên ưu tiên làm rõ yêu cầu với khách hàng ngay lập tức, sử dụng các công cụ quản lý yêu cầu (như Jira hoặc Confluence), và tăng cường kiểm tra để đảm bảo không xảy ra lỗi nghiêm trọng trước khi phát hành.
9. **Dự án phần mềm tồn cơ nhiều lỗi phát triển ở các giai đoạn khác nhau. Làm thế nào để đảm bảo các lỗi phổ biến không tái diễn?**
   Để đảm bảo không tái diễn lỗi, đội có thể áp dụng quy trình kiểm tra nghiêm ngặt hơn, sử dụng các công cụ kiểm tra tự động, tổ chức các buổi phân tích sau dự án (post-mortem) để rút kinh nghiệm, và cải tiến quy trình phát triển.
10. **Mỗi công ty phát triển phần mềm gặp khó khăn trong việc quản lý quy trình do không có chuẩn hóa. Hãy đề xuất giải pháp.**
    Giải pháp bao gồm xây dựng một quy trình chuẩn hóa (như sử dụng mô hình Agile, Scrum, hoặc CMMI), đào tạo nhân viên về quy trình, sử dụng phần mềm quản lý dự án (như Trello, Asana), và thường xuyên đánh giá, cải tiến quy trình.