**Bài tập tổng hợp về Static Testing và Review**

**Phần 1: Lý thuyết**

**Giá trị của Static Testing**

**a. Static Testing là gì? Giải thích giá trị của static testing trong việc phát hiện lỗi trước khi chạy phần mềm.**

Static Testing là quá trình đánh giá các tài liệu, mã nguồn hoặc sản phẩm phần mềm mà không thực thi chương trình. Nó bao gồm các hoạt động như review tài liệu, kiểm tra mã nguồn, và phân tích thiết kế để tìm kiếm lỗi hoặc thiếu sót.

**Giá trị của Static Testing:**

* **Phát hiện lỗi sớm**: Static Testing giúp phát hiện lỗi ngay từ giai đoạn đầu của vòng đời phát triển phần mềm (SDLC), chẳng hạn như trong tài liệu yêu cầu hoặc thiết kế, trước khi mã hóa hoặc thử nghiệm động.
* **Tiết kiệm chi phí**: Sửa lỗi ở giai đoạn sớm (như tài liệu hoặc thiết kế) rẻ hơn nhiều so với sửa lỗi trong giai đoạn thử nghiệm hoặc triển khai.
* **Cải thiện chất lượng**: Giúp đảm bảo rằng các tài liệu và sản phẩm đáp ứng yêu cầu và tiêu chuẩn trước khi tiến hành các bước tiếp theo.
* **Tăng cường hiểu biết chung**: Các buổi review trong static testing thúc đẩy giao tiếp và sự hiểu biết giữa các thành viên trong nhóm và các bên liên quan.
* **Giảm rủi ro**: Phát hiện các lỗi nghiêm trọng như yêu cầu không rõ ràng hoặc thiết kế sai lệch giúp giảm nguy cơ thất bại trong dự án.

**b. Ví dụ cụ thể về lỗi và lý do phát hiện sớm là quan trọng**

**Ví dụ lỗi**: Trong tài liệu yêu cầu của một hệ thống thương mại điện tử, một yêu cầu ghi: "Hệ thống cho phép người dùng thanh toán bằng thẻ tín dụng." Tuy nhiên, tài liệu không đề cập đến việc hỗ trợ các loại thẻ cụ thể (Visa, MasterCard, v.v.) hoặc các yêu cầu bảo mật như chuẩn PCI DSS.

**Tại sao phát hiện sớm quan trọng**:

* **Tránh hiểu lầm**: Nếu không phát hiện sớm, nhóm phát triển có thể triển khai một hệ thống thanh toán không đáp ứng được các loại thẻ phổ biến hoặc không tuân thủ các quy định bảo mật, dẫn đến việc phải sửa đổi mã nguồn hoặc thiết kế lại, gây tốn kém thời gian và chi phí.
* **Đáp ứng kỳ vọng khách hàng**: Phát hiện lỗi này sớm giúp đảm bảo hệ thống đáp ứng đúng nhu cầu của người dùng cuối.
* **Ngăn chặn lỗi lan rộng**: Lỗi trong yêu cầu có thể dẫn đến các lỗi trong thiết kế, mã nguồn, và thử nghiệm, làm tăng chi phí sửa chữa.

**Các loại tài liệu hoặc sản phẩm trong Static Testing**

**a. Liệt kê 5 loại tài liệu/sản phẩm**

1. **Tài liệu yêu cầu phần mềm (SRS)**: Mô tả các chức năng, yêu cầu phi chức năng, và ràng buộc của hệ thống.
2. **Tài liệu thiết kế hệ thống**: Bao gồm sơ đồ kiến trúc, mô hình dữ liệu, và thiết kế giao diện người dùng.
3. **Mã nguồn**: Các tệp mã nguồn (ví dụ: Python, Java, C++) được viết bởi lập trình viên.
4. **Tài liệu kiểm thử**: Bao gồm kế hoạch kiểm thử, kịch bản kiểm thử, và ma trận truy xuất nguồn gốc.
5. **Tài liệu hướng dẫn người dùng**: Tài liệu cung cấp hướng dẫn sử dụng hệ thống cho người dùng cuối.

**b. Ví dụ lỗi tiềm ẩn**

1. **Tài liệu yêu cầu phần mềm**: Yêu cầu không rõ ràng, ví dụ: "Hệ thống phải nhanh" mà không định lượng thời gian phản hồi cụ thể (dưới 2 giây). Static testing có thể phát hiện sự mơ hồ này.
2. **Tài liệu thiết kế hệ thống**: Thiết kế cơ sở dữ liệu thiếu chỉ mục (index) cho các cột thường xuyên truy vấn, gây ra vấn đề về hiệu suất. Static testing giúp phát hiện trước khi triển khai cơ sở dữ liệu.
3. **Mã nguồn**: Một biến được khai báo nhưng không được sử dụng (dead code) trong mã nguồn. Static testing thông qua code review có thể phát hiện và loại bỏ để cải thiện chất lượng mã.
4. **Tài liệu kiểm thử**: Kịch bản kiểm thử bỏ sót trường hợp kiểm tra lỗi (error handling) khi người dùng nhập sai định dạng dữ liệu. Static testing giúp bổ sung các trường hợp kiểm thử bị thiếu.
5. **Tài liệu hướng dẫn người dùng**: Hướng dẫn sử dụng thiếu bước xác nhận email sau khi đăng ký. Static testing phát hiện sự thiếu sót này để đảm bảo tài liệu đầy đủ.

**Quy trình Review**

**a. Các bước chính trong quy trình review**

1. **Lập kế hoạch**: Xác định mục tiêu review, tài liệu/sản phẩm cần review, loại review (walkthrough, inspection), và phân công vai trò.
2. **Chuẩn bị**: Người review đọc trước tài liệu, làm quen với nội dung, và chuẩn bị các câu hỏi hoặc nhận xét ban đầu.
3. **Thực hiện review**: Tổ chức buổi họp review, nơi người review thảo luận các lỗi, vấn đề, hoặc đề xuất cải tiến. Tác giả giải thích hoặc ghi nhận các ý kiến.
4. **Báo cáo và sửa chữa**: Tổng hợp các lỗi/vấn đề được phát hiện, phân loại mức độ nghiêm trọng, và giao cho tác giả để sửa chữa.
5. **Kiểm tra lại (Follow-up)**: Đảm bảo các lỗi đã được sửa chữa và tài liệu/sản phẩm đạt chất lượng mong muốn.

**b. Vai trò và trách nhiệm**

1. **Tác giả (Author)**: Người tạo ra tài liệu hoặc sản phẩm. Trách nhiệm: Chuẩn bị tài liệu, giải thích nội dung trong buổi review, và sửa chữa các lỗi được phát hiện.
2. **Người review (Reviewer)**: Đọc và đánh giá tài liệu/sản phẩm để tìm lỗi hoặc đề xuất cải tiến. Trách nhiệm: Cung cấp phản hồi chi tiết, khách quan, và mang tính xây dựng.
3. **Người điều phối (Moderator)**: Quản lý quá trình review, đảm bảo cuộc họp diễn ra suôn sẻ và hiệu quả. Trách nhiệm: Lập kế hoạch, phân công vai trò, điều hành buổi review, và tổng hợp kết quả.

**c. Cách chọn loại review phù hợp**

* **Walkthrough**: Phù hợp khi mục tiêu là thảo luận và thu thập ý kiến từ nhóm, đặc biệt trong giai đoạn đầu của dự án hoặc khi tài liệu còn sơ khai. Ví dụ: Review tài liệu yêu cầu ban đầu để đảm bảo mọi người hiểu rõ yêu cầu.
* **Inspection**: Phù hợp khi cần kiểm tra chi tiết, chính thức để phát hiện lỗi nghiêm trọng, đặc biệt trong các dự án phức tạp hoặc có yêu cầu cao về chất lượng. Ví dụ: Inspection mã nguồn để đảm bảo tuân thủ tiêu chuẩn lập trình.
* **Lựa chọn dựa trên ngữ cảnh**:
  + **Mục tiêu**: Nếu mục tiêu là kiểm tra tính đúng đắn (correctness), chọn inspection. Nếu cần chia sẻ kiến thức, chọn walkthrough.
  + **Thời gian và nguồn lực**: Walkthrough nhanh hơn và ít tốn kém hơn inspection.
  + **Mức độ quan trọng của tài liệu**: Các tài liệu quan trọng như yêu cầu phần mềm hoặc mã nguồn cần inspection kỹ lưỡng.

**Phản hồi từ Stakeholder**

**a. Tại sao phản hồi từ stakeholder cải thiện chất lượng**

* **Đáp ứng nhu cầu thực tế**: Stakeholder (khách hàng, người dùng cuối) cung cấp góc nhìn thực tế về nhu cầu và kỳ vọng, giúp đảm bảo tài liệu/sản phẩm phù hợp với mục đích sử dụng.
* **Phát hiện thiếu sót sớm**: Phản hồi từ stakeholder giúp phát hiện các lỗi hoặc thiếu sót mà nhóm phát triển có thể bỏ qua, chẳng hạn như chức năng bị thiếu hoặc yêu cầu không rõ ràng.
* **Tăng tính khả dụng**: Phản hồi từ người dùng cuối giúp cải thiện giao diện, trải nghiệm người dùng, và tính dễ sử dụng của sản phẩm.
* **Giảm rủi ro dự án**: Phản hồi thường xuyên giúp điều chỉnh tài liệu/sản phẩm trước khi tiến hành các bước phát triển tốn kém.

**b. Ví dụ về cải thiện tài liệu**

**Ví dụ**: Trong một dự án phát triển ứng dụng đặt vé xem phim, tài liệu yêu cầu ban đầu không đề cập đến việc gửi thông báo xác nhận qua email sau khi đặt vé. Trong buổi review, một stakeholder (đại diện khách hàng) phản hồi rằng người dùng cần nhận thông báo qua email để lưu trữ thông tin vé.

**Cải tiến**: Tài liệu yêu cầu được cập nhật để bao gồm yêu cầu: "Hệ thống gửi email xác nhận chứa thông tin vé (thời gian, địa điểm, mã vé) ngay sau khi đặt vé thành công." Điều này cải thiện trải nghiệm người dùng và giảm nguy cơ khiếu nại từ khách hàng.

**Phần 2: Bài tập thực hành**

**Áp dụng Static Testing**

**a. Xác định lỗi tiềm năng trong yêu cầu**

Yêu cầu: “Hệ thống cho phép người dùng đăng ký sự kiện mà không cần kiểm tra số lượng chỗ trống.”

**Lỗi tiềm năng**:

1. **Thiếu kiểm tra số lượng chỗ trống**: Yêu cầu không đảm bảo rằng hệ thống kiểm tra số lượng chỗ trống, có thể dẫn đến đăng ký quá số lượng cho phép, gây ra tình trạng quá tải hoặc lỗi hệ thống.
2. **Thiếu ràng buộc về xác nhận đăng ký**: Yêu cầu không đề cập đến việc thông báo cho người dùng về trạng thái đăng ký (thành công hoặc thất bại), có thể gây nhầm lẫn hoặc trải nghiệm người dùng kém.

**b. Đề xuất sửa đổi và giải thích**

**Sửa đổi yêu cầu**:

* Yêu cầu mới: “Hệ thống cho phép người dùng đăng ký sự kiện sau khi kiểm tra số lượng chỗ trống còn lại. Nếu số lượng chỗ trống đủ, hệ thống xác nhận đăng ký thành công và gửi thông báo cho người dùng qua email. Nếu không còn chỗ, hệ thống thông báo lỗi rõ ràng cho người dùng.”

**Lý do cải thiện chất lượng**:

* **Ngăn chặn lỗi hệ thống**: Kiểm tra số lượng chỗ trống đảm bảo hệ thống không cho phép đăng ký vượt quá giới hạn, tránh tình trạng quá tải.
* **Cải thiện trải nghiệm người dùng**: Thông báo rõ ràng (thành công hoặc thất bại) giúp người dùng hiểu trạng thái đăng ký, tăng sự hài lòng.
* **Đáp ứng yêu cầu thực tế**: Yêu cầu sửa đổi phản ánh đúng nhu cầu thực tế của người dùng và nhà tổ chức sự kiện.

**Thiết kế quy trình Review**

**Quy trình Review cho tài liệu yêu cầu ứng dụng quản lý sự kiện**

1. **Lập kế hoạch**:
   * Xác định mục tiêu: Đảm bảo tài liệu yêu cầu rõ ràng, đầy đủ, và đáp ứng nhu cầu của nhà tổ chức sự kiện và người dùng.
   * Lựa chọn loại review: Inspection (do yêu cầu cần độ chính xác cao và là tài liệu quan trọng).
   * Phân công vai trò: Tác giả, người review, người điều phối.
2. **Chuẩn bị**:
   * Gửi tài liệu yêu cầu cho các reviewer ít nhất 3 ngày trước buổi review.
   * Các reviewer đọc trước và ghi chú các vấn đề tiềm năng (ví dụ: yêu cầu không rõ ràng, thiếu ràng buộc).
3. **Thực hiện review**:
   * Tổ chức buổi họp inspection, do người điều phối dẫn dắt.
   * Thảo luận từng yêu cầu, tập trung vào tính rõ ràng, tính khả thi, và sự đầy đủ.
   * Ghi nhận các lỗi, ví dụ: yêu cầu không kiểm tra số lượng chỗ trống.
4. **Báo cáo và sửa chữa**:
   * Người điều phối tổng hợp danh sách lỗi, phân loại mức độ nghiêm trọng, và gửi cho tác giả.
   * Tác giả sửa chữa tài liệu dựa trên phản hồi.
5. **Kiểm tra lại**:
   * Người review kiểm tra tài liệu đã sửa để đảm bảo tất cả lỗi được khắc phục.

**Vai trò của các thành viên**

* **Tác giả**: Viết tài liệu yêu cầu, giải thích các yêu cầu trong buổi review, và sửa chữa các lỗi được phát hiện.
* **Người review**: Bao gồm đại diện nhà tổ chức sự kiện, người dùng cuối, và kỹ sư phần mềm. Nhiệm vụ: Đọc tài liệu, phát hiện lỗi, và đề xuất cải tiến.
* **Người điều phối**: Quản lý quy trình review, đảm bảo cuộc họp diễn ra đúng tiến độ và mọi ý kiến được ghi nhận.

**Loại review và lý do lựa chọn**

* **Loại review**: Inspection.
* **Lý do**: Tài liệu yêu cầu là nền tảng cho toàn bộ dự án, đòi hỏi độ chính xác cao. Inspection cho phép kiểm tra chi tiết, phát hiện lỗi nghiêm trọng, và đảm bảo tài liệu đáp ứng tiêu chuẩn chất lượng.

**Phân tích vai trò của phản hồi từ Stakeholder**

**a. Phản hồi từ stakeholder giúp phát hiện vấn đề**

**Tình huống**: Tài liệu thiết kế giao diện của ứng dụng đặt lịch bác sĩ không bao gồm tùy chọn hủy lịch hẹn.

**Cách phản hồi từ stakeholder giúp phát hiện**:

* **Bệnh nhân**: Trong buổi review, bệnh nhân phản hồi rằng họ cần tùy chọn hủy lịch hẹn nếu có thay đổi đột xuất (ví dụ: lịch trình cá nhân thay đổi). Thiếu tính năng này gây bất tiện và làm giảm sự hài lòng.
* **Bác sĩ**: Bác sĩ chỉ ra rằng việc không có tùy chọn hủy lịch hẹn có thể dẫn đến lịch trình không chính xác, gây khó khăn trong việc quản lý thời gian khám bệnh.
* **Phản hồi giúp phát hiện**: Những ý kiến này chỉ ra rằng thiết kế giao diện thiếu một chức năng quan trọng, có thể ảnh hưởng đến trải nghiệm người dùng và hiệu quả hoạt động của hệ thống.

**b. Kế hoạch thu thập và tích hợp phản hồi**

1. **Thu thập phản hồi**:
   * **Khảo sát stakeholder**: Gửi bảng câu hỏi cho bệnh nhân và bác sĩ về các tính năng mong muốn trong giao diện, bao gồm câu hỏi cụ thể về việc hủy lịch hẹn.
   * **Buổi họp review**: Mời đại diện bệnh nhân và bác sĩ tham gia buổi review thiết kế giao diện để thảo luận trực tiếp.
   * **Phỏng vấn nhóm nhỏ**: Tổ chức phỏng vấn với một số bệnh nhân và bác sĩ để hiểu rõ nhu cầu và khó khăn của họ.
2. **Tích hợp phản hồi**:
   * **Cập nhật tài liệu thiết kế**: Thêm tính năng “Hủy lịch hẹn” vào giao diện, bao gồm nút hủy và thông báo xác nhận hủy gửi đến cả bệnh nhân và bác sĩ.
   * **Kiểm tra tính khả thi**: Làm việc với nhóm phát triển để đảm bảo tính năng hủy lịch hẹn có thể được triển khai mà không ảnh hưởng đến hiệu suất hệ thống.
   * **Xác nhận với stakeholder**: Gửi tài liệu thiết kế đã cập nhật cho stakeholder để xác nhận rằng các thay đổi đáp ứng nhu cầu của họ.
3. **Kết quả**: Tài liệu thiết kế giao diện được cải thiện với tính năng hủy lịch hẹn, tăng tính khả dụng và đáp ứng tốt hơn nhu cầu của bệnh nhân và bác sĩ.