Assignment
Software Testing
------o0o------

Nhóm 1

Phan Minh Quân: 21110740

Nguyễn Bảo Quốc: 21110741

Đặng Quang Trường: 21110705

Lê Thành Vinh: 21110940

<u>Assignment 01:</u> Overview of Software Testing

1. **Form: Group discussion, group work.**
2. **Duration: 60 minutes, 10 minutes for submisison.**
3. **Kết quả / Sản phẩm: Word file .DOC or .DOCX**
4. **Requirements:**

Read documents, discuss and answer the following quesitons (brief and concise, i.e NOT MORE THAN 10 lines for each question):

a. Why do we have to perform software testing? What are popular problems that a software might encounter? (Tại sao chúng ta phải thực hiện kiểm thử phần mềm? Những vấn đề cơ bản mà nhiều phần mềm thường gặp phải là gì?)

Chúng ta phải thực hiện kiểm thử phần mềm vì :

- Phần mềm đóng vai trò quan trọng trong nhiều ngành công nghiệp và việc kiểm thử là cần thiết để đảm bảo sản phẩm hoạt động hiệu quả và bảo mật. Kiểm thử giúp phát hiện lỗi sớm, tiết kiệm chi phí, và đảm bảo phần mềm đáp ứng được yêu cầu kinh doanh.

Những vấn đề cơ bản mà nhiều phần mềm thường gặp phải là:

- Tính toán sai: Các phép tính không chính xác dẫn đến kết quả sai.

- Chỉnh sửa dữ liệu không phù hợp: Những thay đổi không chính xác hoặc thiếu hiệu quả trong dữ liệu có thể dẫn đến các lỗi tiềm ẩn.
- Ghép nối và hợp nhất dữ liệu sai: Lỗi khi kết hợp hoặc hợp nhất dữ liệu từ các nguồn khác nhau.
- Kết quả tìm kiếm dữ liệu không chính xác: Tìm kiếm không cho ra kết quả đúng mong đợi.
- Xử lý quan hệ dữ liệu không đúng: Xử lý sai các mối quan hệ giữa các tập dữ liệu.
- Triển khai sai các quy tắc kinh doanh: Lập trình hoặc thực hiện sai các quy tắc nghiệp vụ dẫn đến sai sót.
- Hiệu suất phần mềm không đạt yêu cầu: Phần mềm có thể gặp tình trạng chạy chậm hoặc không đạt được mức hiệu suất mong muốn.
- Dữ liệu gây nhầm lẫn hoặc không chính xác: Dữ liệu hiển thị không đúng hoặc dễ gây nhầm lẫn cho người dùng.
- Phần mềm lỗi thời: Sử dụng công nghệ cũ không còn phù hợp với yêu cầu hiện tại.
- Xử lý không nhất quán: Các quy trình xử lý khác nhau không đồng nhất, gây ra sự không nhất quán trong kết quả.
- Kết quả hoặc hiệu suất không đáng tin cậy: Phần mềm cho ra kết quả không ổn định hoặc không đáng tin cậy.
- Không hỗ trợ đủ cho nhu cầu kinh doanh: Phần mềm không đáp ứng đủ các yêu cầu nghiệp vụ của doanh nghiệp.
- Giao diện không đúng hoặc không đủ với các hệ thống khác: Kết nối hoặc tích hợp với các hệ thống khác không chính xác.
- Kiểm soát bảo mật và hiệu suất chưa đạt yêu cầu: Các biện pháp kiểm soát bảo mật và hiệu suất có thể không đủ mạnh để đảm bảo an toàn và hiệu quả cho hệ thống.
- Xử lý tệp sai: Lỗi khi xử lý các tập tin trong phần mềm.

b. What are the objectives of software testing? What are the main tasks of a Software Tester? (Mục tiêu của việc kiểm thử là gì? Công việc chính của nhân viên kiểm thử phần mềm là gì?)

Mục tiêu của việc kiểm thử:

- Một test case hiệu quả là test case có tiềm năng phát hiện các lỗi chưa từng được nhận ra trước đó
- Một bài kiểm thử thành công là bài kiểm thử phát hiện ra những lỗi mà trước đây chưa từng được nhận biết.
- Một bài kiểm thử hiệu quả không lặp lại những bài kiểm thử khác đã thực hiện trước đó.
- Một bài kiểm thử tốt cần phải nằm trong nhóm những bài kiểm thử xuất sắc nhất.
- Một bài kiểm thử tốt không nên quá dễ dàng mà cũng không nên quá phức tạp.

Công việc chính của một nhân viên kiểm thử phần mềm:

- Phát hiện lỗi càng sớm càng tốt và đảm bảo rằng chúng được khắc phục.
- Hiểu rõ về ứng dụng đang kiểm thử.
- Nghiên cứu chi tiết chức năng để tìm ra những nơi mà lỗi có thể xuất hiện.
- Kiểm tra mã nguồn để đảm bảo rằng từng dòng mã đều được kiểm thử.
- Tạo ra các test case với mục tiêu không chỉ phát hiện các lỗi ẩn mà còn đảm bảo rằng phần mềm có tính sử dụng và đáng tin cậy.

c. Distinguish 2 terms Verification and Validation in Software Testing (Phân biệt 2 thuật ngữ Verification và Validation trong kiểm thử phần mềm.)

Verification: Thường liên quan đến việc xem xét và đánh giá tài liệu, kế hoạch, mã nguồn, yêu cầu và đặc tả. Các hoạt động có thể bao gồm việc sử dụng danh sách kiểm tra, ghi chép các vấn đề, tổ chức thảo luận nhóm, và tiến hành các cuộc họp để đánh giá một cách chi tiết.

Validation: Thường liên quan đến việc kiểm thử thực tế và diễn ra sau khi quá trình Verification hoàn tất. Validation nhằm xác nhận rằng phần mềm thực sự hoạt động theo yêu cầu của người dùng và giải quyết các lỗi tìm thấy trong quá trình kiểm thử.

d. Describe the main phases in a general Software Development Process (Trình bày các công đoạn chính trong 1 quy trình phát triển phần mềm nói chung.)

Các công đoạn chính của một quy trình phát triển phần mềm gồm có: Plan (Lên kế hoạch), Do (Thực hiện kế hoạch), Check (Kiểm tra kết quả), Action (Hành động).

- Plan (P - Lên kế hoạch): Trước hết, ta sẽ xác định rõ ràng mục tiêu mà mình muốn. Đồng thời, xây dựng chiến lược và phương pháp để đạt được mục tiêu đó.
- Do (D - Thực hiện kế hoạch): Sau khi đã có kế hoạch, ta sẽ tạo những điều kiện cần thiết và đào tạo những người tham gia kế hoạch có đủ kiến thức và kỹ năng.
- Check (C - Kiểm tra kế hoạch): Trong quá trình thực hiện kế hoạch, ta sẽ đánh giá xem tiến độ công việc có đúng theo kế hoạch hay không, có đạt được những kết quả như đã đặt ra hay không.
- Action (A - Hành động): Nếu thấy công việc không đúng như những gì mong muốn, hay không đạt được những kết quả như mong đợi, ta thực hiện các biện pháp thay đổi thích hợp.

e. Describe in general the waterfall process and (Rational Unified Process) process. (Trình bày cơ bản về quy trình thác nước và quy trình RUP.)

Quy trình thác nước (Waterfall process): Đây là quy trình phát triển được chia làm các bước mà mỗi bước sẽ phụ thuộc vào kết quả của bước trước đó (theo một cách tuyến tính).

Các bước của quy trình thác nước bao gồm:

- Requirements (Thu thập yêu cầu)
- Analysis and Design (Phân tích và thiết kế hệ thống)
- Implementation (Triển khai)
- Testing (Kiểm thử)
- Operation and deployment (Vận hành và deploy)
- Maintenance (Bảo trì)

Quy trình thác nước phù hợp với những dự án phần mềm đã có những yêu cầu rõ ràng từ ban đầu và không có sự thay đổi nhiều trong quá trình phát triển. Không hợp với những dự án có sự thay đổi liên tục về yêu cầu, dự án phức tạp vì quy trình thiếu sự linh hoạt.

Quy trình RUP (Rational Unified Process): Đây là quy trình phát triển có cấu trúc, là một khung (framework) linh hoạt được kết hợp từ nhiều mô hình phát triển khác nhau.

Các giai đoạn chính của RUP bao gồm:

- Inception (Khởi đầu)

- Elaboration (Làm rõ)
- Constructor (Xây dựng)
- Transition (Chuyển tiếp)
- Production (Sản phẩm)

Quy trình RUP phù hợp với những dự án lớn, phức tạp; các dự án phát triển phần mềm hướng đối tượng hay sử dụng UML; những dự án cần quản lí rủi ro. Tuy nhiên quy trình cần đội ngũ phát triển và quản lý dự án có trình độ cao, không phù hợp với những dự án nhỏ vì có thể gây ra sự phức tạp không cần thiết.

f. Describe the V-model in testing, then list basic types of testing for each phase. (Trình bày về mô hình kiểm thử hình chữ V, qua đó nêu ngắn gọn các loại kiểm thử cơ bản ứng với từng công đoạn phát triển.)

Mô hình kiểm thử hình chữ V thuộc dạng SDLC (Software Development Life Cycle), là một phần mở rộng của mô hình thác nước (Waterfall) nhưng nghiêm ngặt hơn. Trong mô hình V, việc kiểm thử sẽ chạy song song với quá trình phát triển.

Một số loại kiểm thử cơ bản ứng với từng công đoạn phát triển:

- Requirement Analysis (Phân tích yêu cầu) sẽ có quá trình tương ứng là Acceptance Testing (Kiểm thử chấp thuận).
- System Design (Thiết kế hệ thống) sẽ có quá trình tương ứng là System Testing (Kiểm thử hệ thống).
- Architecture Design (Thiết kế kiến trúc hệ thống) thì sẽ là Integration Testing (Kiểm thử tích hợp).
- Module Design (Thiết kế mô-đun) thì sẽ có quá trình tương ứng là Unit Testing (Kiểm thử đơn vị).
- Coding thì việc test sẽ do các nhà phát triển thực hiện.

g. Describe the main phases in the software testing process (Trình bày các bước chính trong quy trình kiểm thử phần mềm.)

Trong quy trình kiểm thử phần mềm có 6 giai đoạn:

- Phân tích yêu cầu (Requirement Analysis): Mục tiêu của giai đoạn này là: Làm rõ các yêu cầu chức năng và phi chức năng của phần mềm. Trong giai đoạn này, nhóm kiểm thử sẽ thường xuyên xem xét tài liệu yêu cầu và làm việc với các bên liên quan để thu thập các thông tin về chức năng và phi chức năng (hiệu suất, tính bảo mật, tính khả dụng) từ đó làm rõ các yêu cầu của phần mềm.
- Lập kế hoạch kiểm thử (Test Planning): Trong giai đoạn này, nhóm kiểm thử sẽ tạo kế hoạch kiểm thử chi tiết, bao gồm chiến lược kiểm thử, các loại kiểm thử sẽ thực hiện (chức năng, hiệu suất, bảo mật,..), nguồn lực cần thiết, công cụ kiểm thử và lịch trình kiểm thử. Mục tiêu là xác định phạm vi, cách tiếp cận, tài nguyên và lịch trình kiểm thử.
- Phát triển kịch bản kiểm thử (Test Case Development): Viết các trường hợp kiểm thử cụ thể (test case) dựa trên yêu cầu, mô tả các bước cần thực hiện, dữ liệu đầu vào, kết quả mong đợi và các điều kiện tiên quyết. Mục tiêu là xác định các trường hợp kiểm thử cần thực hiện.
- Thiết lập môi trường kiểm thử (Environment Setup): Cài đặt và cấu hình các hệ thống cần thiết, như máy chủ, cơ sở dữ liệu, công cụ tự động hóa kiểm thử, và chuẩn bị dữ liệu kiểm thử. Mục tiêu là chuẩn bị môi trường kiểm thử (phần cứng, phần mềm, mạng, công cụ kiểm thử,...) để kiểm thử có thể tiến hành suôn sẻ.
- Thực hiện kiểm thử (Test Execution): Chạy các trường hợp kiểm thử bằng cách thủ công hoặc sử dụng công cụ tự động hóa. Ghi lại kết quả kiểm thử, so sánh với kết quả mong đợi, và ghi nhận các lỗi.
- Kết thúc chu kỳ kiểm thử (Test Cycle Closure): Giai đoạn này nhóm phát triển sẽ xem xét và phân tích kết quả của quá trình kiểm thử, đưa ra báo cáo kiểm thử, xác định các mục tiêu đã đạt được và không đạt được, đánh giá phạm vi kiểm tra và đánh giá phần mềm trước khi bàn giao sản phẩm.

h. State the basice principles in software testing (Nêu các nguyên lý cơ bản trong kiểm thử phần mềm.)

Trong kiểm thử phần mềm có 7 nguyên lý cơ bản:

- Kiểm thử chỉ ra sự hiện diện của lỗi (Testing shows the presence of defects): Kiểm thử phần mềm nhằm mục đích phát hiện các lỗi trong phần mềm. Kiểm thử phần mềm không

chứng minh rằng phần mềm không có lỗi. Ngay cả khi không tìm thấy lỗi nào, cũng không thể chứng minh phần mềm 100% không có lỗi. Kiểm thử chỉ làm giảm số lượng lỗi nhưng không thể loại bỏ hoàn toàn.

- Kiểm thử toàn diện là không thể (Exhaustive testing is impossible): Không thể kiểm thử tất cả các kịch bản hoặc đầu vào có thể có của phần mềm, đặc biệt là đối với các hệ thống phức tạp. Nếu kiểm thử tất cả trường hợp của phần mềm sẽ mất rất nhiều chi phí, công sức hơn giá trị của phần mềm mang lại, điều này sẽ không thực tế. Chính vì vậy, việc lựa chọn các tình huống kiểm thử là rất quan trọng.
- Kiểm thử sớm (Early testing): Việc bắt đầu kiểm thử từ các giai đoạn sớm của vòng đời phát triển phần mềm (như giai đoạn yêu cầu và thiết kế) có thể giúp phát hiện sớm các lỗi, từ đó giảm chi phí sửa lỗi.
- Cụm lỗi (Defect clustering): Trong thực tế, phần lớn các lỗi thường tập trung vào một số ít các mô-đun hoặc thành phần của phần mềm. Điều này có nghĩa là việc kiểm thử kỹ lưỡng các khu vực dễ bị lỗi sẽ hiệu quả hơn.
- Nghịch lý thuốc trừ sâu (Pesticide paradox): Khi chúng ta lặp đi lặp lại các trường hợp kiểm thử sẽ đến 1 lúc nào đó việc này sẽ không còn hiệu quả trong việc phát hiện lỗi mới. Chính vì vậy, cần phải thường xuyên xem xét và thay đổi các trường hợp kiểm thử.
- Kiểm thử phụ thuộc vào ngữ cảnh (Testing is context-dependent): Việc kiểm thử cần phải xem xét và đưa ra các phương pháp khác nhau đối với mỗi ngữ cảnh cụ thể của dự án, phần mềm. Ví dụ như kiểm thử ứng dụng thương mại sẽ khác ứng dụng giáo dục.
- Sự sai lầm về việc không có lỗi (Absence-of-errors fallacy): Việc phát hiện và sửa lỗi không đảm bảo rằng phần mềm đáp ứng đúng yêu cầu của người dùng hoặc khách hàng. Do đó, kiểm thử cần phải đảm bảo rằng phần mềm không chỉ không có lỗi mà còn phải đáp ứng các yêu cầu và mong đợi của người dùng.

i. What are the limitations/disadvantages of software testing (Những hạn chế của việc kiểm thử là gì?)

Những hạn chế của kiểm thử:

- Phạm vi kiểm thử là không đầy đủ: Đối với các ứng dụng lớn và phức tạp, việc xây dựng trường hợp sử dụng, các kịch bản có thể xảy ra một cách toàn diện là điều khó có thể

thực hiện. Thông thường kiểm thử sẽ chỉ kiểm tra các trường hợp, kịch bản tổng quát và toàn diện.

- Hạn chế về thời gian và nguồn lực: Quá trình kiểm thử đòi hỏi nhiều thời gian, đặc biệt là khi phạm vi kiểm thử lớn hoặc khi cần kiểm thử thủ công. Điều này có thể dẫn đến chi phí tăng cao cho dự án.
- Không thể phát hiện tất cả lỗi: Không thể đảm bảo rằng kiểm thử sẽ phát hiện được tất cả các lỗi trong phần mềm. Một số lỗi có thể không xuất hiện trong môi trường kiểm thử nhưng lại xuất hiện khi phần mềm được sử dụng trong thực tế.
- Phụ thuộc vào chất lượng của test case: Kết quả kiểm thử phụ thuộc rất nhiều vào chất lượng của các test case. Nếu các test case không được thiết kế tốt, chúng có thể không phát hiện được những lỗi quan trọng.
- Phụ thuộc vào môi trường: Môi trường kiểm thử có thể không hoàn toàn giống với môi trường thực tế mà phần mềm sẽ chạy. Điều này có thể dẫn đến các vấn đề không được phát hiện trong quá trình kiểm thử.
- Khó khăn trong kiểm thử tự động: Đối với một số loại kiểm thử, đặc biệt là kiểm thử giao diện người dùng (UI) hoặc kiểm thử chức năng phức tạp, việc tự động hóa có thể rất khó khăn và tốn nhiều công sức để duy trì.

j. What are test cases? State the basic principles in test case design. (Test case là gì? Nêu các nguyên tắc cơ bản khi thiết kế test case.)

- Test case được hiểu là tập hợp các trường hợp thử nghiệm để kiểm tra xem chức năng có hoạt động đúng yêu cầu hay chưa. Một Test case sẽ mô tả cụ thể các giá trị đầu vào, các hành động hoặc sự kiện diễn ra một cách tuần tự, kèm theo đó là tập kết quả mong đợi.
- Các nguyên tắc cơ bản khi thiết kế test case:
- Mô tả rõ ràng, cụ thể về dữ liệu đầu vào
- Mô tả rõ ràng, cụ thể về dữ liệu đầu ra mong đợi
- Phải thiết kế test case ở cả hai trường hợp: Dữ liệu hợp lệ và dữ liệu không hợp lệ
- Nên tránh việc viết test case cho các module mà chính mình tạo ra để hạn chế ý kiến chủ quan, dẫn đến bỏ qua những lỗi tìm tàng
- Phải thiết kế số lượng test case đủ nhiều để đảm bảo bao phủ nhiều tình huống sẽ có thể xảy ra

k. Describe briefly about blackbox testing and whitebox testing. (Trình bày ngắn gọn về các kỹ thuật kiểm thử hộp đen và kiểm thử hộp trắng).

- Kiểm thử hộp đen là tập trung kiểm tra dựa trên dữ liệu đầu vào và dữ liệu đầu ra nhằm so sánh độ chính xác của module đó. Không cần quan tâm logic, giải thuật được sử dụng bên trong mà chỉ cần quan tâm kết quả cuối cùng mà nó mang lại.
- Kiểm thử hộp trắng là quá trình kiểm tra giải thuật, logic, cấu trúc dữ liệu, luồng vận hành của dữ liệu khi chức năng đó được thực hiện nhằm xác định tính chính xác của một module cụ thể.

5. References:

[1] Lecture Chapter 1: Overview of Software Testing

[2] Part 2: Software Testing Fundamental.

[3] Chapter 1&2, Nguyen Van Hiep lectures.

[4] Other Internet và wiki resources.

[5] FITA Academy, Why do we need software testing? https://www.fitaacademy.in/blog/why-do-we-need-software-testing/

[6] Geeksforgeeks, SDLC V-Model – Software Engineering https://www.geeksforgeeks.org/software-engineering-sdlc-v-model/