THỰC HÀNH MODULE 01

Họ và tên: Dương Thái Bảo

Mã số sinh viên: 21037621

Bài 1. Trả lời các câu hỏi dưới:

a. Đưa ra một số lý do giải thích tại sao đặc tả là nơi chứa nhiều bug nhất.

- Đặc tả là nơi chứa nhiều bug vì:
    - Khi thu thập yêu cầu bằng lời nói, ngôn ngữ có thể gây ra hiểu lầm.
    - Khách hàng không nhất quán trong yêu cầu về sản phẩm.
    - Không hiểu yêu cầu
    - Yêu cầu về sản phẩm bị thay đổi liên tục
    - Khách hàng đặc tả thiếu yêu cầu
    - Bất đồng trong ngôn ngữ.
    - Người thu thập yêu cầu không cẩn thận, không kiểm tra lại yêu cầu.

⇨ Những yếu tố liên quan đến con người thường phức tạp và có nhiều vấn đề khi làm việc cùng nhau giữa các bên, có thể gây ra nhiều vấn đề.

b. Giải thích tại sao chi phí sửa bug sau khi sản phẩm release lại cao hơn khi mới bắt đầu.

- Khi một sản phẩm được release mà phát hiện ra lỗi, ta sẽ cần phải kiểm tra lại rất nhiều công đoạn trước đó, bao gồm: lỗi trong quá trình xây dựng, lỗi trong thiết kế để phù hợp với yêu cầu, và thậm chí phải kiểm tra lại yêu cầu.
- Điều đó sẽ gây ra một số vấn đề như:
    - Tốn thêm chi phí về thời gian để kiểm tra xem lõi từ đâu
    - Tốn nhân lực để xử lý lỗi
    - Tốn thêm tiền bạc để giải quyết vấn đề.
    - Nếu lỗi xuất phát từ thu thập sai yêu cầu, có thể mất nhiều thời gian hơn nữa vì lỗi có thể sẽ không được nhìn thấy vởi team IT và thậm chí cả team kiểm thử => Gây mất thời gian và chi phí về nhân lực.
- Vì vậy, việc tìm lỗi và kiểm tra tính chính xác trong quá trình phát triển là rất quan trọng.

c. Nêu và giải thích 7 nguyên tắc của kiểm thử.

Các nguyên tắc:

(1) Cho thấy sự hiện diện của lỗi:

- Kiểm thử có thể cho thấy sự hiện diện của lỗi, nhưng không thể chứng minh rằng nó không có lỗi.
- Kiểm thử làm giảm xác xuất của việc không tìm thấy lỗi tồn tại trong phần mềm, kể cả nếu không tìm thấy lỗi, không có nghĩa là mọi thứ chính xác.

(2) Kiểm thử tất cả là bất khả thi:

- Kiểm tra tất cả mọi thử (từ các input và các điều kiện) là gần như không thể/

    - Ví dụ như: bạn có 10 field cần test, trong đó có 5 giá trị hợp lệ, số lượng test sẽ là 5^10 = 9.765.625 trường hợp.
- Thay vì kiểm thử toàn bộ, tập trung vào phân tích các mối nguy hiểm và ưu tiên, sử dụng mối nguy hiểm để xác định:
    - Kiểm thử gì trước
    - Kiểm thử gì nhiều hơn
    - Cái nào không cần kiểm thử
- Ưu tiên như thế nào? Có thể đánh giá theo điều kiện (dựa trên các mối nguy hiểm)
    - Kiểm thử cái dễ lỗi nghiêm trọng nhất
    - Kiểm thử cái dễ lỗi khi hiển thị
    - Kiểm thử cái có vẻ dễ lỗi
    - Hỏi khách hàng về độ ưu tiên trong yêu cầu
    - Cái quan trọng nhất trong doanh nghiệp của khách hàng
    - Phần thay đổi thường xuyên nhất
    - Phần có nhiều vấn đề nhất trong quá khứ
    - Phần phức tạp nhất, hoặc công nghệ quan trọng nhất

(3) Kiểm thử sớm nhất có thể

- Hoạt động kiểm thử nên được bắt đầu sớm nhất có thể trong phần mềm hoặc vòng đời phát triển của hệ thống, và có thể tập trung vào các mục tiêu được xác định.
    - Lỗi được tìm thấy trễ trong quá trình phát triển có thể tốn nhiều chi phí để giải quyết.
        - VD: Một lỗi trong thông số kỹ thuật của sản phẩm thì có thể dễ sửa, nhưng nếu lỗi đó được chuyển sang dạng mã thì việc sửa lỗi có thể tốn chi phí và thời gian.

(4) Tập trung vào phân cụm

- Một số nhỏ các module có thể chứa nhiều lỗi:
    - Các lỗi thường có xu hướng tập trung quanh một khu vực hoặc một hàm (“điểm nóng”)
        - Quy tắc 80-20: Khoảng 80% các vấn đề được tìm thấy trong 20% các module.
    - Bằng cách nhận diện và tập trung vào các cụm này, người kiểm thử có thể đạt hiệu quả trong kiểm thử các vùng nhạy cảm trong khi vẫn có thể kiểm thử đồng thời các phần còn lại

(5) Hiệu ứng thuốc trừ sâu

- Nếu các bộ kiểm thử lặp đi, lặp lại, cuối cùng thì bộ kiểm thử đó sẽ không còn tìm thấy lỗi nào nữa.
- Để khắc phục nghịch lý này, các bộ kiểm thử phải được xem xét, sửa đổi thường xuyên và làm mới. Ngoài ra các bộ kiểm thử nên được viết lại để thực thi trên các phần khác nhau của phần mềm hoặc hệ thống để có khả năng tìm ra nhiều lỗi hơn.

(6) Kiểm thử phục thuộc vào ngữ cảnh

- Kiểm thử có thể hoàn tất khác nhau phụ thuộc vào ngữ cảnh.
    - Bộ kiểm thử như sau có thể không được áp dngj cho toàn bộ, vì các phần mềm khác nhau có yêu cầu, tính năng và mục đích khác nhau
        - Ví dụ: Một phần mềm cần tính bảo mật cao (như quân đội) thì sẽ cần kiểm thử khác với cửa hàng thương mại.
    - Không phải tất cả phần mềm để có cùng một mức độ nguy hiểm, và không phải tất cả vấn đề đều có cùng độ ảnh hưởng khi nó xảy ra sự cố.
        - Ví dụ: Một giao diện bị lỗi chính tả, điều này có quan trọng không:
            - Nếu là trang web cá nhân của tôi
            - Nếu là trang web của công ty

(7) Ngụy biện không có lỗi

- Nếu chúng ta không tìm thấy lỗi, đâu có nghĩa người dùng sẽ chấp nhận phần mềm của chúng ta đúng không?
- Tìm và sửa lỗi sẽ không giúp ích nếu hệ thống xây dựng không sử dụng được, và cũng không đúng những gì người dùng cần và mong đợi.
    - Kiểm thử không tìm thấy lỗi, khác với việc kết luận rằng phần mềm không có lỗi.
    - Người kiểm thử nên cho rằng tất cả phần mềm đều có lỗi, thậm chí là lỗi đó bị ẩn đi.

Bài 2. Hãy tìm các defect/bug (càng nhiều càng tốt) trong hình dưới đây. Các lỗi có thể là giao diện không nhất quán, lỗi chính tả, trùng lắp, …

Mô tả lỗi:

(1) Lỗi chính tả: Dư dấu)

(2) Bị lệch so với các chữ khác

(3) Chữ more không đồng nhất (More)

(4) Bị mất viền

(5) Bị mất viền

(6) Không căn chính giữa, chỗ đặt nút không hợp lý

(7) Lỗi lặp từ

(8) Bị lặp nút

(9) Không đồng nhất cách phân cách nút

(10) Không hiện favico web

(11) Nút X bị mất một đoạn nhỏ

(12) Trang đã load xong nhưng vẫn quay

(13) Không căn giữa trang

(14) Nút xổ menu bị ngược.

(15) Lỗi chính tả (Google)

(16) Lỗi chính tả (Maps – Google Maps)

(17) Khoảng cách không đều so với các phần khác

(18) Không đồng nhất cách ghi (In hoa chữ đầu mỗi từ)

(19) Không đồng nhất cách ghi (In hoa chữ đầu của từ)

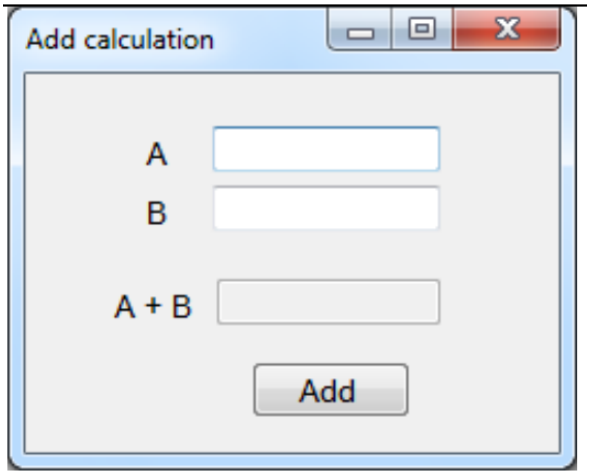
(20) Không đồng nhất cách phân cách nút

(21) Viền không đồng nhất (Hơi vàng hơn nút bên trái)

Bài 3. Hãy chỉ ra ít nhất một lỗi bất kỳ trong ứng dụng hoặc website bạn đã từng sử dụng

Lỗi không thấy bản dịch “Không có tin nhắn”

Bài 4. Tìm các trường hợp kiểm thử (test case) cho chương trình cộng 2 số nguyên có giao diện sau:



STT	Dữ liệu nhập	Kết quả mong muốn
1	A: 0 B: 0	0
2	A: 1 B: 2	3
3	A: 2 B: 1	3
4	A: -1 B: 0	-1
5	A: 0 B: -1	-1
6	A: 1 B: 0	1
7	A: 0 B: 1	1
8	A: 1 B: Để trống	Phải điền số
9	A: Để trống B: 1	Phải điền số
10	A: Đế trống B: Để trống	Phải điền số
11	A: abc B: 1	Kiểu dữ liệu không hợp lệ
12	A: 1 B: abc	Kiểu dữ liệu không hợp lệ
13	A: 1.2 B: 2	Yêu cầu nhập số nguyên

14	A: 2 B: 1.2	Yêu cầu nhập số nguyên
15	A: 2.1 B: 2.1	Yêu cầu nhập số nguyên

Bài 5. Cho chương trình đọc một loạt các số đo nhiệt độ tuỳ ý (số nguyên) trong khoảng -60°C đến +60°C và in ra giá trị trung bình của các số này. Tìm các trường hợp cần kiểm thử cho chương trình này, lập theo bảng sau:

STT	Dữ liệu nhập	Kết quả mong muốn
1	20; 20; 30; 40; 60	34
2	-61; 20; 30; 20;10	Lỗi khoảng
3	20; 30; 40; 61	Lỗi khoảng
4	-60; -20; -20; -30; -40	-34
5	0; 0; 0; 0; 0; 0; 0	0
6	-10; -20; -50; 10; 20; 30	3.33
7	10.3; 10	Yêu cầu nhập số nguyên
8	-10.3; 10	Yêu cầu nhập số nguyên
9	Abc; 10; 30	Kiểu dữ liệu không hợp lệ