DEMO Nhom18 Đề tài 13.Exploratory testing

Đảm bảo chất lượng và kiểm thử phần mềm (Trường Đại học Công nghiệp Thành phố Hồ Chí Minh)

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

Đề tài: 13.Exploratory testing (Kiểm thử thăm dò)

Giảng viên hướng dẫn	**: Châu Thị Bảo Hà**
Sinh viên thực hiện	**: Trần Trọng Tín**
Môn học	**: Đảm bảo chất lượng và Kiểm thử phần mềm**
Lớp học phần	**: DHKTPM16CTT**
MSSV	**: 20094991**
Nhóm	**: 18**

TP.HCM, ngày ….. tháng ….. năm …..

Lời giới thiệu

Xin chào mọi người, Trong đề tài này, tôi sẽ nghiên cứu về phương pháp kiểm thử Exploratory testing. Đây là một phương pháp kiểm thử linh hoạt và tương tác, cho phép nhóm kiểm thử tìm hiểu và khám phá các lỗ hổng và vấn đề trong phần mềm. Exploratory testing không chỉ đơn thuần là việc kiểm thử ngẫu nhiên mà nó yêu cầu sự sáng tạo và tư duy phản biện từ phía nhóm kiểm thử.

Phương pháp này cho phép nhóm kiểm thử tự do khám phá và kiểm tra các tính năng và chức năng của phần mềm một cách tự nhiên, giúp phát hiện ra các lỗi mà kiểm thử kiểm soát không thể phát hiện được. Trong quá trình nghiên cứu, tôi sẽ tìm hiểu về các phương pháp và kỹ thuật thực hiện Exploratory testing, bao gồm việc xác định phạm vi kiểm thử, lập kế hoạch và triển khai kiểm thử, đồng thời cũng tìm hiểu về các công cụ hỗ trợ và quy trình kiểm thử liên quan. Tôi tin rằng việc nghiên cứu về Exploratory testing sẽ mang lại nhiều kiến thức bổ ích và ứng dụng trong công việc kiểm thử phần mềm. Hy vọng rằng đề tài này sẽ giúp tôi và mọi người hiểu rõ hơn về phương pháp kiểm thử này và áp dụng nó một cách hiệu quả trong công việc của chúng ta.

Thuật ngữ "exploratory testing", được đặt ra bởi Cem Kaner trong cuốn sách Kiểm tra phần mềm máy tính, đề cập đến một cách tiếp cận để kiểm thử rất khác với kiểm thử theo kịch bản. Thay vì một kiểm tra tuần tự các yêu cầu, tiếp theo là thiết kế và tài liệu kiểm tra các trường hợp, tiếp theo là việc thực hiện các trường hợp kiểm thử đó, thử nghiệm thăm dò, theo định nghĩa của James Bach, là "học tập đồng thời, thiết kế thử nghiệm và thực hiện thử nghiệm." Người thử nghiệm thiết kế và thực hiện các thử nghiệm trong khi khám phá sản phẩm.

Mục lục

Lời giới thiệu
Mục lục
Bảng phân công công việc
Chương I : Khái niệm về kiểm thử thăm dò “Exploratory testing”
1. Tổng quan về kiểm thử phần mềm :
1.1. Exploratory testing là gì :
1.2. Định nghĩa “Exploratory testing” :
1.3. Khả năng “Exploratory testing” :
2. Đặc điểm “Exploratory testing” :
2.1. Sự khác nhau giữa thử nghiệm theo kịch bản và kiểm thử thăm dò :
3. Một số khái niệm liên quan :
4. Quy định kiểm thử :
Chương II : Cách thức hoạt động của “Exploratory testing”
1. cách thức khởi chạy Exploratory testing :
2. Quy trình kiểm thử thăm dò :
3. Cách thức thực hiện kiểm thử :
3.1. Tạo một nguyên tắc phân loại lỗi (phân loại)
3.2. Test charter: Điều lệ thử nghiệm
3.3. Timebox: Hộp thời gian
3.4. Đánh giá kết quả
3.5. Trao đổi
4. Phương pháp thực hiện kiểm thử :
4.1. Dự đoán :
4.2. Sơ đồ kiến trúc và trường hợp sử dụng :
4.3. Khiếm khuyết quá khứ :
4.4. Xử lý lỗi :
4.5. Thảo luận :
4.6 Câu hỏi và danh sách kiểm tra :
Chương III. Ưu điểm và nhược điểm của “Exploratory testing”
1. Các trường hợp sử dụng Exploratory testing và lợi ích khi sử dụng :
1.1. Trường hợp sử dụng Exploratory testing :
1.2. Lợi ích mà Exploratory testing mang lại cho người dùng :
2. Uu điểm của Exploratory testing :
3. Nhược điểm của Exploratory testing :
4. Một vài lợi ích khi sử dụng Exploratory testing trong mô hình agile :
4.1. Quản lý thời gian

4.2. Tìm ra lỗi quan trọng
4.3. Đưa ra các trường hợp kiểm thử hiệu quả
4.5. Có lợi trong trường hợp yêu cầu thay đổi nhanh chóng
4.6. Tương thích với thời gian ngắn của phương pháp Scrum
5. Điểm mấu chốt và điều lệ của Exploratory testing :
5.1. Điểm mấu chốt :
5.2. Điều lệ :
6. Thách thức :
Chương IV. Áp dụng thực tiễn
1. Áp dụng kiểm thử thăm dò vào dự án :
1.1. Cách áp dụng Exploratory testing vào trong dự án :
1.2. các bước để áp dụng Exploratory testing :
1.2.1. Phân loại bug :
1.2.2 Điều lệ kiểm thử (Test charter) :
1.2.3 Thời gian quy định :
1.2.4 Đánh giá kết quả :
1.2.5 Thảo luận kết quả :
2. Phương pháp thực hiện Exploratory testing một cách hiệu quả :
2.1. Đoán lỗi :
2.2. Tạo sơ đồ kiểm thử và các trường hợp sử dụng :
2.3. Những lỗi trước đó :
2.4. Xử lý lỗi :
2.5. Thảo luận :
2.6. Câu hỏi và checklist :
3. Một số lưu ý khi sử dụng Exploratory testing :
3.1. Tập trung vào mục tiêu :
3.2. Không tạo kịch bản, nhưng cần lập kế hoạch kiểm thử :
3.3. Đừng cố kiểm tra quá nhiều :
3.4. Lưu giữ một bản ghi rõ ràng về những việc đã làm :
3.5. Xác định phạm vi (thời gian, quy trình) :
3.6. Chọn các kỹ thuật thăm dò đáp ứng với nhu cầu của dự án :
4. Một vài ví dụ khi sử dụng Exploratory testing :
4.1. Ví dụ 1 :
4.2. Ví dụ 2 :
Chương V. Kết luận
1. Tổng kết :
2. Một số hình ảnh về Cem Kaner :
3. Tài liệu kham khảo :

Thành viên nhóm

STT	Họ và tên	Lớp	Mã số sinh viên
1	Trần Trọng Tín	DHKTPM16FTT	20094991

Bảng phân công công việc

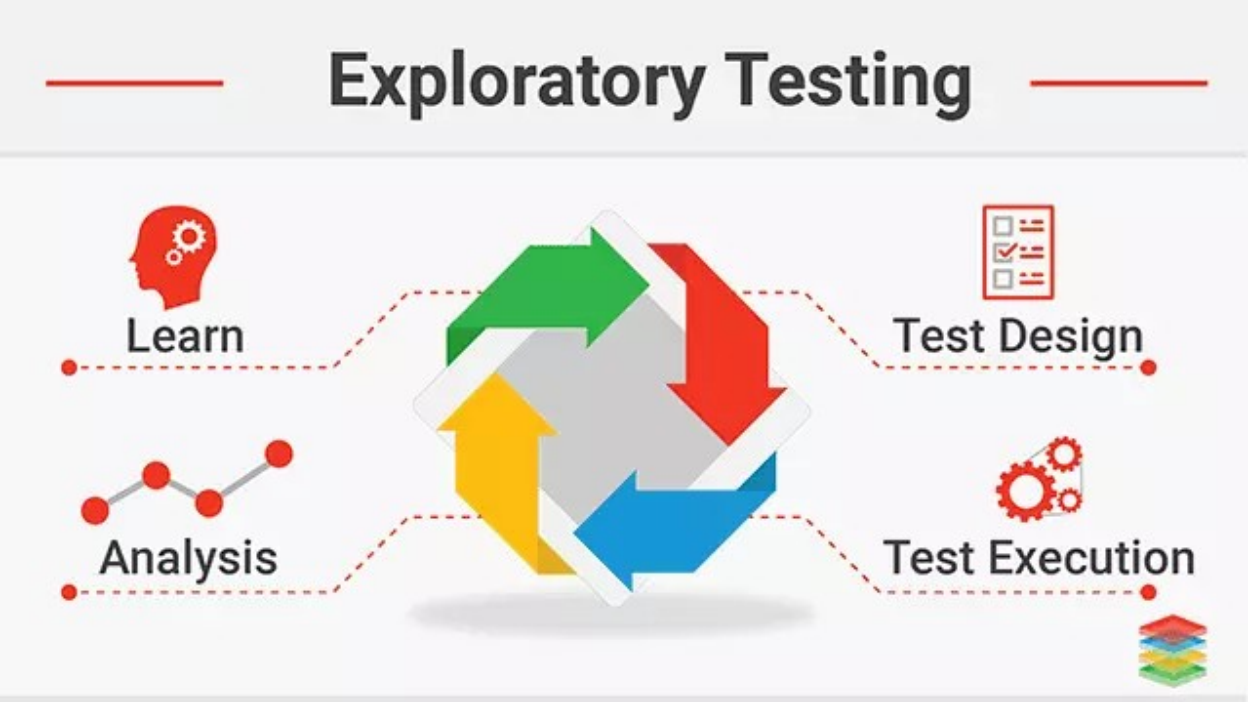
STT	Ngày	Công việc thực hiện	Người thực hiện
1	Từ 04/09/2023 Đến 08/09/2023	Nhận đề tài từ giảng viên và đăng kí đề tài: 13.Exploratory testing	Trần Trọng Tín
2	Từ 11/09/2023 Đến 17/09/2023	Kham khảo nguồn thông tin từ các trang mạng và giáo trình giảng viên hướng dẫn	Trần Trọng Tín
3	Từ 18/09/2023 Đến 24/09/2023	Thu thập thông tin tìm được và làm chương I và chương II	Trần Trọng Tín
4	Từ 25/09/2023 Đến 01/10/2023	Chỉnh sửa thông tin từ ý kiến và trợ giúp của giảng viên	Trần Trọng Tín
5	Từ 02/09/2023 Đến 08/10/2023	Thu thập thông tin tìm được và làm chương III và chương VI	Trần Trọng Tín
6	Từ 09/09/2023 Đến 12/10/2023	Hoàn thành chương V và chỉnh sửa theo hướng dẫn của giảng viên	Trần Trọng Tín
7	Từ 13/09/2023 Đến 15/10/2023	Nộp đề tài cho giảng viên và tiếp thu thêm kiến thức	Trần Trọng Tín

Chương I : Khái niệm về kiểm thử thăm dò “Exploratory testing”

1. Tổng quan về kiểm thử phần mềm :

1.1. Exploratory testing là gì :

- Exploratory testing là một loại kiểm thử phần mềm trong đó các trường hợp kiểm thử không được tạo trước nhưng người kiểm tra có thể kiểm tra hệ thống thông tin một cách nhanh chóng. Họ có thể ghi lại những ý tưởng về những gì cần kiểm tra trước khi thực hiện kiểm tra.
- Trong kiểm thử thăm dò, tester kiểm soát thiết kế các trường hợp kiểm thử khi chúng được thực hiện thay vì ngày, tuần hoặc thậm chí vài tháng trước đó. Ngoài ra, thông tin mà tester thu được từ việc thực hiện một bộ kiểm thử sau đó hướng dẫn tester thiết kế và thực hiện bộ kiểm thử tiếp theo.
- Kiểm thử thăm dò là rất quan trọng bất cứ khi nào việc chọn trường hợp kiểm thử tiếp theo sẽ được chạy không thể được xác định trước mà nên được chọn dựa trên các thử nghiệm trước đó và kết quả của chúng.



Hình 1. Kiểm thử thăm dò (Exploratory testing)

1.2. Định nghĩa “Exploratory testing” :

- Kiểm thử thăm dò được định nghĩa là "học tập đồng thời, thiết kế thử nghiệm và thực hiện kiểm thử". Người thử nghiệm thiết kế và thực hiện các thử nghiệm trong khi khám phá sản phẩm.
- Định nghĩa của exploratory testing ra đời từ năm 1984 với cách hiểu là một kiểu thử nghiệm phần mềm nhấn mạnh sự tự do cá nhân và trách nhiệm của người thử

nghiệm để liên tục tối ưu hóa chất lượng công việc của mình bằng cách xử lý việc học liên quan đến thử nghiệm, thiết kế thử nghiệm, thực hiện thực nghiệm và giải thích kết quả thực nghiệm, tất cả đề là các hoạt động hỗ trợ lẫn nhau chạy song song trong suốt dự án.

- "Exploratory Testing là cách tiếp cận quá trình test cho phép bạn áp dụng năng lực, kỹ năng và kỹ xảo của người kiểm thử (QA) một cách hữu hiệu nhất". Đầu tiên những nhân viên kiểm thử phần mềm (QA) phải hiểu về ứng dụng đó bằng việc khám phá nó dựa trên sự hiểu biết về việc chúng xảy ra với các kịch bản kiểm thử nào. Sau đó bắt đầu quá trình kiểm tra thực tế của ứng dụng.

1.3. Khả năng "Exploratory testing" :

- Exploratory testing nhấn mạnh đến tự do cá nhân và trách nhiệm của mình trong thử nghiệm vì người thử nghiệm được kiểm tra và học hỏi lặp đi lặp lại trong suốt quá trình không tìm kiếm một câu trả lời được thiết kế sẵn cũng không thể đi chệch hướng. Điều này không có nghĩa là nó không thiểu sự chuẩn bị mà là không hạn chế người thử nghiệm.

- Exploratory testing giúp người kiểm tra khám phá một ứng dụng để xác định và ghi lại các lỗi tiềm ẩn của nó. Người thử nghiệm bắt tay vào một quá trình điều tra và khám phá để thử nghiệm hiệu quả một sản phẩm. Ngoài ra exploratory testing còn được hiểu là một cách để tìm hiểu về ứng dụng và thiết kế các trường hợp thử hiện chức năng và quy hồi sẽ được thực hiện trong tương lai.

2. Đặc điểm "Exploratory testing" :

Kiểm thử thăm dò khác so với các loại kiểm thử có kịch bản khác (nó cần có khả năng tư duy, đào sâu và khám phá cao hơn)

- Có thể xem kiểm thử có kịch bản (Scripted testing) như một đoàn tàu, nó sẽ chạy trên một đường ray có sẵn (Test case).

- Với kiểm thử thăm dò, nó như một chiếc ô tô, cũng là từ điểm A đi đến B, nhưng sẽ không nhất định theo một cung đường, mà có thể chạy những đoạn đường khác nhau để đến đúng B.

Kiểm thử thăm dò được áp dụng phổ biến trong mô hình Agile, vì:

- Nó đem lại phản hồi nhanh chóng về hệ thống cho đội phát triển

- Đề cao sự khác biệt trong quan điểm test của tất cả các vai trò, thành viên trong team dự án

- Nâng cao sự khám phá, điều tra và hiểu biết sâu sắc hơn phần mềm đang kiểm thử.

Hình 2. Đặc điểm "Exploratory testing"

2.1. So sánh sự khác nhau :

2.1.1. So sánh Test Scenario và Exploratory testing :

TEST SCENARIO	EXPLORATORY TESTING
Đưa ra từ yêu cầu	Đưa ra từ yêu cầu và khám phá trong quá trình thử nghiệm
Xác định các trường hợp thử nghiệm trước khi quá trình thử nghiệm diễn ra	Xác định các trường hợp thử nghiệm trong quá trình thử nghiệm
Mục đích để xác nhận thử nghiệm so với các yêu cầu	Mục đích để điều tra, tìm hiểu hệ thống hoặc ứng dụng
Nhấn mạnh dự đoán và ra quyết định	Nhấn mạnh khả năng thích ứng và học tập
Liên quan đến thử nghiệm đã được xác nhận	Liên quan đến điều tra
Tập trung kiểm soát thử nghiệm	Tập trung cải tiến thiết kế thử nghiệm
Kiểm soát bởi kịch bản	Kiểm soát bởi tâm trí của người kiểm thử

2.1.2. So sánh Ad-Hoc Testing và Exploratory testing :

AD-HOC TESTING	EXPLORATORY TESTING
Ad-hoc testing bắt đầu khi đã tìm hiểu ứng dụng và sau đó mới thực hiện quá trình kiểm tra thực tế.	Exploratory Testing sẽ tìm hiểm ứng dụng trong khi thực hiện test
Trong Ad-hoc testing tài liệu không phải là nhu cầu cần thiết. Đội QA tham gia vào quá trình kiểm tra mà không cần tài liệu đặc tả yêu cầu.	Trong Exploratory Testing tài liệu là bắt buộc. Để đảm bảo về chất lượng của dự án, tài liệu chi tiết của quá trình kiểm tra là cần thiết.
Ad-hoc quan tâm đến sự hoàn hảo của sản phẩm.	Exploratory Testing quan tâm đến việc khảo sát chất lượng sản phẩm hơn là hiểu về sản phẩm.
Ad-hoc là công nghệ test của ứng dụng, nó cung cấp vai trò quan trọng trong việc sản xuất phần mềm.	Tester (QA) trước hết cần phải biết một chức năng phần mềm. Trước khi thực hiện kiểm tra toàn bộ ứng dụng hoặc phần mềm tester (QA) cần phải tìm hiểu nó thông qua Exploratory Testing.
Thử nghiệm này thực thi một lần duy nhất. Tester (QA) kiểm thử nó một lần tại một thời điểm, tuy nhiên nếu có bất kỳ vấn đề gì tìm thấy trong quá trình test thì cần phải thực hiện lặp lại thao tác.	Đây là phương pháp thử nghiệm kết hợp các kết quả kiểm tra trong quá trình nghiên cứu và việc tạo ra một giải pháp mới.
Nó chủ yếu hoạt động trên các mối quan tâm về nghiệp vụ và làm gia tăng sự hiểu biết về các ứng dụng.	Nó phân loại các vấn đề và so sánh chúng từ các vấn đề được tìm thấy trong quá khứ. Điều này giúp làm giảm thời gian cho việc kiểm tra.

3. Một số khái niệm liên quan :

- Chất lượng phần mềm (Software quality): là mức độ mà một hệ thống, thành phần hay quy trình đáp ứng các yêu cầu của đặc tả phần mềm, các nhu cầu mong đợi của khách hàng hoặc người sử dụng .
- Đảm bảo chất lượng phần mềm (Software quality assurance): là một quy trình có kế hoạch và hệ thống của tất cả các hành động cần thiết để cung cấp các thông tin đầy đủ để đảm bảo các sản phẩm có phù hợp với các yêu cầu về kỹ thuật hay không.

- Xác nhận (Validation): là quá trình đánh giá một hệ thống hay cấu phần trong hay cuối của quá trình phát triển để xác định xem nó đáp ứng yêu cầu quy định.
- Xác minh, kiểm chứng (Verification): là quá trình đánh giá một hệ thống hay thành phần để xác định xem các sản phẩm của một giai đoạn phát triển nhất định đáp ứng các điều kiện áp đặt tại lúc bắt đầu của giai đoạn đó .
- Lỗi (Error): Lỗi là những vấn đề mà con người mắc phải trong quá trình phát triển các sản phẩm phần mềm.
- Sai (Fault): Sai là kết quả của lỗi, hay nói khác đi, lỗi sẽ dẫn đến sai.
- Thất bại (Failure): Thất bại xuất hiện khi một lỗi được thực thi.

4. Quy định kiểm thử :

- Trong thử nghiệm thăm dò, người thử nghiệm kiểm soát việc thiết kế các trường hợp thử nghiệm khi chúng được thực hiện thay vì ngày, tuần hoặc thậm chí vài tháng trước đó. Thông tin mà người kiểm tra thu được từ việc thực hiện một bộ kiểm tra sau đó hướng dẫn người kiểm tra thiết kế và thực hiện bộ kiểm tra tiếp theo. Thử nghiệm thăm dò là việc khám phá, tìm hiểu về phần mềm, nó làm được gì, không làm được gì, cái gì hiệu quả và cái gì không. Người kiểm tra liên tục đưa ra quyết định về những gì cần kiểm tra tiếp theo và nơi dành thời gian (có giới hạn). Đây là cách tiếp cận hữu ích nhất khi không có hoặc có thông số kỹ thuật kém và khi thời gian bị hạn chế nghiêm trọng.

- Quy trình thử nghiệm thăm dò bao gồm:
    - Đưa ra giả thuyết về hoạt động đúng đắn của hệ thống. Thiết kế một hoặc nhiều thử nghiệm có thể bác bỏ giả thuyết.
    - Tiến hành thí nghiệm và quan sát kết quả.
    - Đánh giá kết quả theo giả thuyết.
    - Lặp lại quá trình này cho đến khi giả thuyết được chứng minh hoặc không được chấp nhận.
- Thử nghiệm thăm dò được sử dụng khi bạn được yêu cầu phát hiện lỗi và bạn được yêu cầu cung cấp phản hồi nhanh chóng về chất lượng sản phẩm trong thời gian ngắn, hữu ích cho thử nghiệm theo kịch bản.

Chương II : Cách thức hoạt động của “Exploratory testing”

1. cách thức khởi chạy Exploratory testing :

- Exploratory testing không quá phức tạp để chạy bằng máy mà nó có thể được chạy bang tay. Exploratory testing có thể được mở và kết thúc, yêu cầu người thử nghiệm xác định cách thức và những gì cần tự kiểm tra. Trong exploratory testing – thử nghiệm thăm dò bao gồm thử nghiệm thăm dò dựa trên chiến lược và thử nghiệm khám phá dựa trên kịch bản. Cả hai loại thử nghiệm này đều yêu cầu người thử nghiệm tập trung vào các khu vực cụ thể hoặc luồng người dùng trong ứng dụng.
- Người thử nghiệm khám phá phải đặt mình ở cảm nhận của người dùng để có thể dự đoán cách họ sẽ hành xử. Khi chạy thử nghiệm thăm dò người kiểm tra hoặc nhà phát triển QA Tester có kinh nghiệm thường được yêu cầu để thử nghiệm thăm dò.

2. Quy trình kiểm thử thăm dò :

- Trong thử nghiệm thăm dò, người thử nghiệm kiểm soát việc thiết kế các trường hợp thử nghiệm khi chúng được thực hiện thay vì ngày, tuần hoặc thậm chí vài tháng trước đó. Thông tin mà người kiểm tra thu được từ việc thực hiện một bộ kiểm tra sau đó hướng dẫn người kiểm tra thiết kế và thực hiện bộ kiểm tra tiếp theo.
- Thử nghiệm thăm dò là việc khám phá, tìm hiểu về phần mềm, nó làm được gì, không làm được gì, cái gì hiệu quả và cái gì không. Người kiểm tra liên tục đưa ra quyết định về những gì cần kiểm tra tiếp theo và nơi dành thời gian (có giới hạn). Đây là cách tiếp cận hữu ích nhất khi không có hoặc có thông số kỹ thuật kém và khi thời gian bị hạn chế nghiêm trọng.

3. Cách thức thực hiện kiểm thử :

Chuẩn bị kiểm thử khám phá trải qua 5 giai đoạn chi tiết dưới đây và nó còn được gọi là quản lý kiểm tra dựa trên phiên:

3.1. Tạo một nguyên tắc phân loại lỗi (phân loại)

Việc phân loại này dựa vào các yếu tố sau:

- Phân loại các loại lỗi phổ biến được tìm thấy trong các dự án trước đây
- Phân tích nguyên nhân gốc phân tích các vấn đề hoặc lỗi
- Tìm các rủi ro và phát triển ý tưởng để kiểm tra ứng dụng.

3.2. Test charter: Điều lệ thử nghiệm

- Điều lệ thử nghiệm nên đề nghị gồm: Kiểm thử những gì; Làm thế nào để kiểm tra chúng;Những gì cần phải được xem xét

- Ý tưởng thử nghiệm là điểm khởi đầu của kiểm thử khám phá
- Điều lệ thử nghiệm giúp xác định cách người dùng cuối có thể sử dụng hệ thống

3.3. Timebox: Hộp thời gian

- Phương pháp này bao gồm một cặp người thử nghiệm làm việc cùng nhau không dưới 90 phút
- Không nên có bất kỳ thời gian bị gián đoạn trong phiên 90 phút đó
- Timebox có thể được kéo dài hoặc giảm 45 phút
- Phiên này khuyến khích người kiểm tra phản ứng và phản hồi từ hệ thống và chuẩn bị cho kết quả chính xác

3.4. Đánh giá kết quả

- Đánh giá các khuyết tật
- Rút ra bài học từ thử nghiệm
- Phân tích vùng phủ sóng

3.5. Trao đổi

- Tổng hợp kết quả đầu ra
- So sánh kết quả với điều lệ
- Kiểm tra xem có cần thử nghiệm bổ sung nào không

Điều quan trọng của cách thức thực hiện là lập tài liệu theo dõi những điều sau đây:

- Phạm vi kiểm tra - Cho dù chúng tôi đã ghi chú về phạm vi bảo hiểm của các trường hợp kiểm tra và cải thiện chất lượng của phần mềm
- Rủi ro - Những rủi ro nào cần được bảo hiểm và đó là những rủi ro quan trọng nào?
- Nhật ký thực hiện kiểm tra - Bản ghi về thực hiện kiểm tra
- Các vấn đề / Truy vấn - Ghi chú về câu hỏi và các vấn đề trên hệ thống

4. Phương pháp thực hiện kiểm thử :

4.1. Dự đoán :

- Đoán được sử dụng để tìm một phần của chương trình có khả năng có nhiều lỗi hơn. Kinh nghiệm trước đây khi làm việc trên một sản phẩm / phần mềm / công nghệ tương tự giúp dự đoán tốt hơn

4.2. Sơ đồ kiến trúc và trường hợp sử dụng :

- Sơ đồ kiến trúc mô tả các tương tác và mối quan hệ giữa các thành phần và mô-đun khác nhau. Các trường hợp sử dụng cung cấp cho sự hiểu biết về việc sử dụng sản phẩm từ quan điểm của người dùng cuối. Kỹ thuật thăm dò có thể sử dụng các sơ đồ này và các trường hợp sử dụng để kiểm tra sản phẩm.

4.3. Khiếm khuyết quá khứ :

- Nghiên cứu các khiếm khuyết được báo cáo trong các bản phát hành trước giúp hiểu được các tính năng của phần mềm được dự kiến sẽ có các khiếm khuyết tối đa.

4.4. Xử lý lỗi :

- Xử lý lỗi là một phần của mã hóa thực hiện các hành động thích hợp trong trường hợp có lỗi. Kiểm thử khám phá có thể được thực hiện bằng các kịch bản khác nhau để kiểm tra xử lý lỗi.

4.5. Thảo luận :

- Kiểm thử khám phá cũng có thể được lên kế hoạch dựa trên sự hiểu biết về phần mềm trong các cuộc thảo luận và họp dự án.

4.6 Câu hỏi và danh sách kiểm tra :

- Những câu hỏi như thế là gì, khi nào, như thế nào, ai và tại sao có thể cung cấp manh mối cho việc kiểm tra phần mềm khám phá.

Chương III. Ưu điểm và nhược điểm của “Exploratory testing”

Exploratory testing thích hợp sử dụng trong các trường hợp nào ?

- Khi ứng dụng không có tài liệu đặc tả yêu cầu hoặc không có tài liệu tối thiểu ch-việc kiểm thử (testplan, checklist, test design,…)
- Khi bạn muốn hoàn thành việc kiểm thử trong thời gian ngắn
- Khi bạn phải kiểm thử ứng dụng sớm trong một chu kỳ phát triển phần mềm

1. Các trường hợp sử dụng Exploratory testing và lợi ích khi sử dụng :

1.1. Trường hợp sử dụng Exploratory testing :

- Trong chu trình phát triển của một tính năng hoặc ứng dụng riêng lẻ, thử nghiệm khám phá thường được thực hiện khi một tính năng hoặc ứng dụng sắp hoàn thành để người kiểm tra có thể có được cái nhìn chính xác hơn về cách ứng dụng sẽ hoạt động trong sản xuất. Đối với một số nhóm chuyển động nhanh, thử nghiệm thăm dò được sử dụng trước đó trong quá trình phát triển để kiểm tra các vấn đề và hồi quy mới sau mỗi đoạn mã mới được xuất xưởng.
- Kiểm tra thăm dò không phải lúc nào cũng cung cấp thời gian sử dụng hiệu quả nhất nhưng lại có những tình huống thực hiện trường hợp kiểm thử và kiểm tra hồi

quy có ý nghĩa hơn. Chẳng hạn nếu một công ty phát hành một chức năng mới mà không liên quan đến hàng tồn kho hoặc tiền tệ thì việc sử dụng người kiểm tra theo cách chức năng không phát huy tác dụng. Các thử nghiệm hồi quy đơn giản có thể được tiến hành để đảm bảo rằng các tính năng khác không bị vô tình ảnh hưởng bởi các tính năng mới được phát hành ở nơi khác.

1.2. Lợi ích mà Exploratory testing mang lại cho người dùng :

- Exploratory testing thường được sử dụng như một hình thức kiểm tra khả năng sử dụng bởi vì bản chất tự do và sáng tạo của kiểm tra chạy sao chép chặt chẽ hơn trải nghiệm người dùng cho các hoạt động nhất định.

- Lợi ích chính mà thử nghiệm khám phá là đòi hỏi công việc chuẩn bị tối thiểu cho những người thử nghiệm cho phép họ nhanh chóng đi sâu vào một tính năng và đánh giá chất lượng của nó, cung cấp cho nhóm phản hồi nhanh về cách di chuyển tính năng về trước. Sử dụng exploratory testing hỗ trợ rất nhiều cho các tester để bên cạnh việc kiểm tra họ có thể đưa ra thêm nhiều quyết định có thể làm.

- Một lợi ích khác từ exploratory testing là giúp người kiểm tra có thể sử dụng lý luận suy diễn dựa trên kết quả trước đó để hướng dẫn thử nghiệm trong tương lai một cách nhanh chóng. Họ không cần phải hoàn thành một loạt các bài kiểm tra theo kịch bản hiện tại trước khi tập trung hoặc chuyển sang khám phá một môi trường giàu mục tiêu hơn. Điều này cũng tăng tốc phát hiện lỗi khi được áp dụng hợp lý.

2. Uu điểm của Exploratory testing :

- Khuyến khích được sự sáng tạo và trực giác của người kiểm tra

- Tạo ra ý tưởng mới trong quá trình thực hiện thử nghiệm

- Kiểm thử thăm dò có giá trị trong các tình huống mà việc chọn trường hợp kiểm thử tiếp theo sẽ chạy không thể được xác định trước, nhưng nên dựa trên các thử nghiệm trước đó và kết quả của chúng.

- Thử nghiệm thăm dò rất hữu ích khi bạn được yêu cầu cung cấp phản hồi nhanh về chất lượng sản phẩm trong thời gian ngắn, với ít thời gian, ngoài đầu bạn, khi các yêu cầu mơ hồ hoặc thậm chí không tồn tại, hoặc sớm trong quá trình phát triển khi hệ thống có thể không ổn định.

- Thử nghiệm thăm dò rất hữu ích khi phát hiện lỗi, chúng tôi muốn khám phá kích thước, phạm vi và biến thể của lỗi đó để cung cấp phản hồi tốt hơn cho các nhà phát triển của chúng tôi.

- Kiểm thử thăm dò là một bổ sung hữu ích cho kiểm thử theo kịch bản khi các bài kiểm tra theo kịch bản trở nên "mệt mỏi", nghĩa là chúng không phát hiện ra nhiều lỗi.

3. Nhược điểm của Exploratory testing :

- Thử nghiệm thăm dò không có khả năng ngăn ngừa các khuyết tật. Bởi vì việc thiết kế các trường hợp kiểm thử theo kịch bản bắt đầu trong giai đoạn thu thập yêu cầu và thiết kế, các lỗi có thể được xác định và sửa chữa sớm hơn.

- Nếu bạn đã chắc chắn chính xác những thử nghiệm nào phải được thực hiện và theo thứ tự nào, thì không cần phải khám phá. Viết và sau đó thực hiện các bài kiểm tra theo kịch bản.

- Nếu bạn được yêu cầu bởi hợp đồng, quy tắc hoặc quy định để sử dụng thử nghiệm theo kịch bản thì hãy làm như vậy. Cân nhắc thêm các bài kiểm tra thăm dò như một kỹ thuật bổ sung.

4. Một vài lợi ích khi sử dụng Exploratory testing trong mô hình agile :

4.1. Quản lý thời gian

- Như đã đề cập ở trên, mô hình Agile có thời hạn chặt chẽ và cả nhóm chỉ có vài tuần để hoàn thành mọi thứ, điều rất quan trọng là phải hiểu và kiểm tra (đặc biệt là tất cả các tính năng mới được thêm vào) trong khung thời gian đó. Exploratory testing cung cấp một cách tốt hơn để đối phó với thời hạn đó.

4.2. Tìm ra lỗi quan trọng

- Khi bạn lặp đi lặp lại việc release một phần của dự án mỗi tháng hoặc vài tuần một lần, bạn đã tự động hóa một số test cases hồi quy (regression test cases), một số test cases chức năng (functional test cases) và một số test cases mà trước đó chỉ thực hiện được bằng việc kiểm thử thủ công (test manual). Chúng hữu ích trong trường hợp khi các tính năng đã tồn tại, nếu trong lần lặp này, có một số tính năng mới được thêm vào ứng dụng thì bạn sẽ cần phải hiểu chúng đúng cách rồi viết các test cases cho nó và cuối cùng, tự động hóa chúng.

4.3. Đưa ra các trường hợp kiểm thử hiệu quả

- Đây là sự tiếp nối trước đó của các lỗi quan trọng được tìm thấy bằng cách sử dụng exploratory testing. Khi bạn nói đã tìm thấy 10 lỗi trong tính năng mới, bạn có thể ghi lại từng trường hợp và viết test case của những lỗi đó. Đây sẽ là tập hợp các test cases chủ yếu tập trung vào tính năng mới và đủ hiệu quả để tìm các lỗi quan trọng.

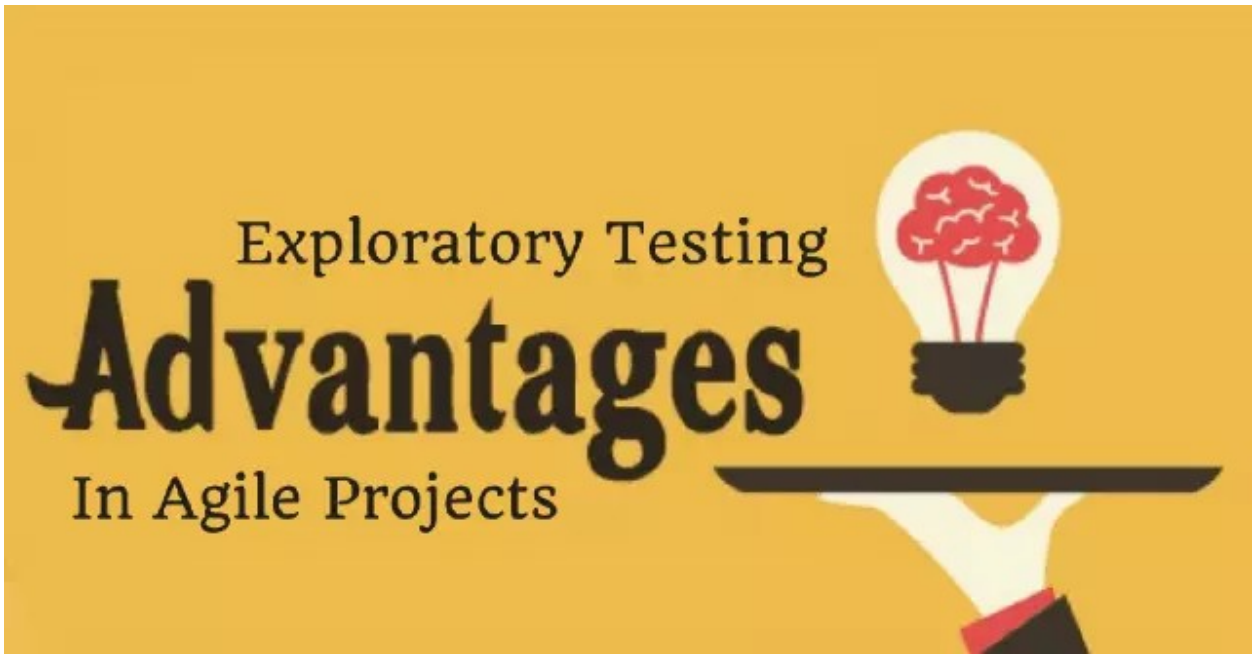
4.5. Có lợi trong trường hợp yêu cầu thay đổi nhanh chóng

- Mô hình Agile không thể đoán trước khi chúng ta nói về các yêu cầu. Khách hàng có thể thay đổi yêu cầu của mình bất cứ lúc nào, đó là lý do tại sao Agile linh hoạt để sử dụng và phát triển. Nhưng đối với đội kiểm thử, nó trở thành một thách thức để kiểm tra các yêu cầu thay đổi ở giữa vòng đời phát triển một cách hiệu quả. Trong những trường hợp như vậy, exploratory testing mang lại lợi ích to lớn khi bạn không có thời gian để lên kế hoạch cho các test case của mình vì thời gian gấp

rút để xác định các yêu cầu mới cần kiểm thử. Nếu tự tin, bạn có thể kiểm tra các trường hợp quan trọng nhất và có thể yên tâm rằng ứng dụng ổn định

4.6. Tương thích với thời gian ngắn của phương pháp Scrum

- Như chúng ta biết rằng sprint trong phương pháp Scrum có thời gian rất nhỏ, từ hai tuần đến một tháng hoặc đôi khi ít hơn thế. Trong trường hợp này, exploratory testing là một lợi thế lớn để tìm ra các lỗi trong các thay đổi mới nhanh nhất có thể, giúp nhóm lập trình có thể dễ dàng sửa chữa và toàn bộ nhóm Scrum có thể thực hiện kịp trong thời hạn sprint đó rất hiệu quả.



Hình 3. Một vài lợi ích khi sử dụng Exploratory testing trong mô hình agile

5. Điểm mấu chốt và điều lệ của Exploratory testing :

5.1. Điểm mấu chốt :

- Điều lệ tập trung nỗ lực của người thử nghiệm thăm dò trong hộp thời gian. Các điều lệ có thể bao gồm:
    + Điều tra kỹ lưỡng một chức năng hệ thống cụ thể
    + Xác định và sau đó kiểm tra quy trình công việc của hệ thống
    + Xác định và xác minh tất cả các tuyên bố được đưa ra trong hướng dẫn sử dụng
    + Hiểu các đặc tính hiệu suất của phần mềm
    + Đảm bảo rằng tất cả các trường đầu vào được xác thực đúng cách

+ Buộc tất cả các điều kiện lỗi để xác minh từng thông báo lỗi

+ Kiểm tra thiết kế theo tiêu chuẩn giao diện người dùng

5.2. Điều lệ :

- Có thể thực hiện thử nghiệm thăm dò mà không cần điều lệ. Đây được gọi là "thử nghiệm thăm dò tự do". Trong quá trình này, người kiểm thử sử dụng các kỹ năng của họ đến mức tối đa khi họ đồng thời tìm hiểu sản phẩm và thiết kế và thực hiện các bài kiểm tra.

- Người kiểm tra thăm dò là những người kiểm tra lành nghề. (Tất nhiên, chúng tôi muốn tester có kỹ năng bất kể chúng tôi đang sử dụng quy trình kiểm thử nào!) Phương pháp kiểm tra thăm dò tôn trọng những kỹ năng đó và trên thực tế, phụ thuộc vào chúng. Những người thử nghiệm thăm dò tốt là:

    + Các nhà mô hình hóa tốt, có thể tạo ra các mô hình tinh thần của hệ thống và hành vi đúng đắn của nó.

    + Người quan sát cẩn thận, có thể nhìn, nghe, đọc và hiểu.

    + Người thiết kế kiểm thử lành nghề, có thể lựa chọn kỹ thuật thiết kế thử nghiệm phù hợp trong từng tình huống. Bach nhấn mạnh, "Một người thử nghiệm thăm dò trước hết là một nhà thiết kế thử nghiệm."

    + Có thể đánh giá rủi ro và để nó hướng dẫn thử nghiệm của họ.

    + Các nhà tư tưởng phản biện, có thể tạo ra các ý tưởng đa dạng, tích hợp các quan sát, kỹ năng và kinh nghiệm của họ để đồng thời khám phá sản phẩm, thiết kế các bài kiểm tra và thực hiện các bài kiểm tra.

    + Các phóng viên cẩn thận, có thể báo cáo nghiêm ngặt và hiệu quả cho người khác những gì họ đã quan sát.

    + Tự quản lý, có thể dẫn đầu trong thử nghiệm hơn là thực hiện một kế hoạch do người khác nghĩ ra.

    + Không bị phân tâm bởi những vấn đề tầm thường.

6. Thách thức :

Có rất nhiều thách thức của kiểm thử thăm dò sẽ được trình bày chi tiết dưới đây:

- Phải học sử dụng app và phần mềm

- Phải nhân rộng lỗi

- Xác định được tools cần sử dụng

- Xác định được test case tốt nhất để sử dụng

- Báo cáo lại kết quả trong khi không có kịch bản/ case nào để so sánh cùng với kết quả hiện tại và output
- Không biết khi nào sẽ dừng test bởi vì nó không define được test case cần thực hiện

Chương IV. Áp dụng thực tiễn

1. Áp dụng kiểm thử thăm dò vào dự án :

1.1. Cách áp dụng Exploratory testing vào trong dự án :

- Trong giai đoạn đầu của dự án, khi mà có rất nhiều dự thay đổi của yêu cầu hoặc quá trình phát triển, việc sử dụng thử nghiệm thăm dò sẽ mang lại hiệu quả cao.
- Các developer cũng có thể áp dụng kiểm thử thăm dò vào unit testing, nó sẽ có hiệu quả tối ưu trong quá trình làm quen với hệ thống.
- Trong mô hình Agile, có có chu kỳ ngắn và thời gian để viết Testcase bị hạn chế.
- Thực hiện kiểm thử thăm dò giúp tiết kiệm thời gian, có các output đầu ra quan trọng từ các chu kỳ cũ, được áp dụng cho các scrum tiếp theo.

1.2. các bước để áp dụng Exploratory testing :

Kiểm thử thăm dò được thực hiện thông qua 5 bước, nó còn được gọi là phiên quản lý dựa trên thử nghiệm (SBTM cycle).

1.2.1. Phân loại bug :

- Phân loại các lỗi thường gặp trong các dự án trước đây
- Phân tích nguyên nhân gốc rễ của vấn đề hoặc lỗi
- Tìm ra những rủi ro và phát triển ý tưởng kiểm thử ứng dụng.

1.2.2 Điều lệ kiểm thử (Test charter) :

- Là một tuyên bố của các mục tiêu kiểm thử, có thể thử nghiệm các ý tưởng về làm thế nào để kiểm tra.

Điều lệ kiểm thử bao gồm:

- Test cái gì
- Test như thế nào
- Cần xem xét, chú ý cái gì trong quá trình test

- Những ý tưởng kiểm thử là điểm bắt đầu của kiểm thử thăm dò
- Điều lệ kiểm thử sẽ xác định người dùng cuối có thể sử dụng hệ thống hay không.

1.2.3 Thời gian quy định :

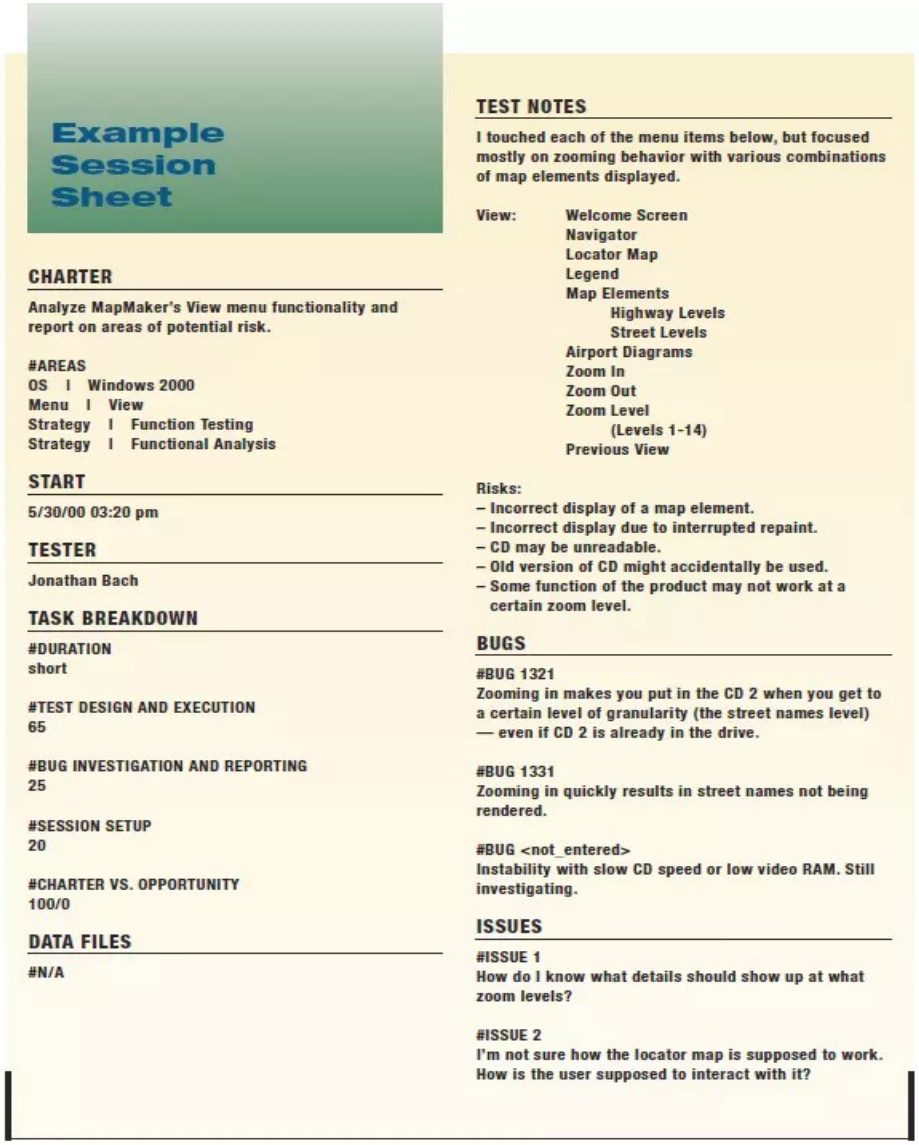
- Thời gian cho một cặp tester làm việc cùng nhau không ít hơn 90 phút
- Không nên có bất kỳ gián đoạn trong 90 phút làm việc đó.
- Khung thời gian có thể được giãn ra hoặc giảm đi 45 phút.
- Mỗi phiên làm việc nên được kiểm thử về trạng thái phản hồi của hệ thống và có chuẩn bị đầu ra một cách chính xác.

1.2.4 Đánh giá kết quả :

- Đánh giá các lỗi
- Rút ra bài học từ kiểm thử
- Phân tích độ bao phủ

1.2.5 Thảo luận kết quả :

- Tổng hợp kết quả sau khi test
- So sánh kết quả test với điều lệ (test charter)
- Kiểm tra xem có cần phải kiểm thử bổ sung hay không (đánh giá dựa vào lượng bug của hệ thống, một feature nào đó có thể có clustering bug).



Example Session Sheet

CHARTER

Analyze MapMaker's View menu functionality and report on areas of potential risk.

#AREAS
OS | Windows 2000
Menu | View
Strategy | Function Testing
Strategy | Functional Analysis

START

5/30/00 03:20 pm

TESTER

Jonathan Bach

TASK BREAKDOWN

#DURATION
short

#TEST DESIGN AND EXECUTION
65

#BUG INVESTIGATION AND REPORTING
25

#SESSION SETUP
20

#CHARTER VS. OPPORTUNITY
100/0

DATA FILES

#N/A

TEST NOTES

I touched each of the menu items below, but focused mostly on zooming behavior with various combinations of map elements displayed.

View:
- Welcome Screen
- Navigator
- Locator Map
- Legend
- Map Elements
  - Highway Levels
  - Street Levels
- Airport Diagrams
- Zoom In
- Zoom Out
- Zoom Level
  - (Levels 1-14)
- Previous View

Risks:
- Incorrect display of a map element.
- Incorrect display due to interrupted repaint.
- CD may be unreadable.
- Old version of CD might accidentally be used.
- Some function of the product may not work at a certain zoom level.

BUGS

#BUG 1321
Zooming in makes you put in the CD 2 when you get to a certain level of granularity (the street names level) — even if CD 2 is already in the drive.

#BUG 1331
Zooming in quickly results in street names not being rendered.

#BUG <not_entered>
Instability with slow CD speed or low video RAM. Still investigating.

ISSUES

#ISSUE 1
How do I know what details should show up at what zoom levels?

#ISSUE 2
I'm not sure how the locator map is supposed to work. How is the user supposed to interact with it?

Hình 4. form mẫu áp dụng Exploratory testing

2. Phương pháp thực hiện Exploratory testing một cách hiệu quả :

2.1. Đoán lỗi :

Dựa vào những kiến thức, kinh nghiệm từ các dự án đã từng làm và tổng quan về dự án, bạn có thể đưa ra dự đoán về một số tính năng của hệ thống có thể có lỗi.

2.2. Tạo sơ đồ kiểm thử và các trường hợp sử dụng :

Nên tạo các sơ đồ về tương tác và mối liên quan giữa các chức năng trong hệ thống sẽ giúp cover được các case nhiều hơn. Và tạo các user case giúp hiểu được cách sử dụng sản phẩm từ góc độ người dùng cuối.

2.3. Những lỗi trước đó :

Nghiên cứu các lỗi đã từng xảy ra ở các bản build trước đó để có thể tập trung vào các tính năng có thể xảy ra các lỗi tương tự hoặc có nhiều lỗi hơn.

2.4. Xử lý lỗi :

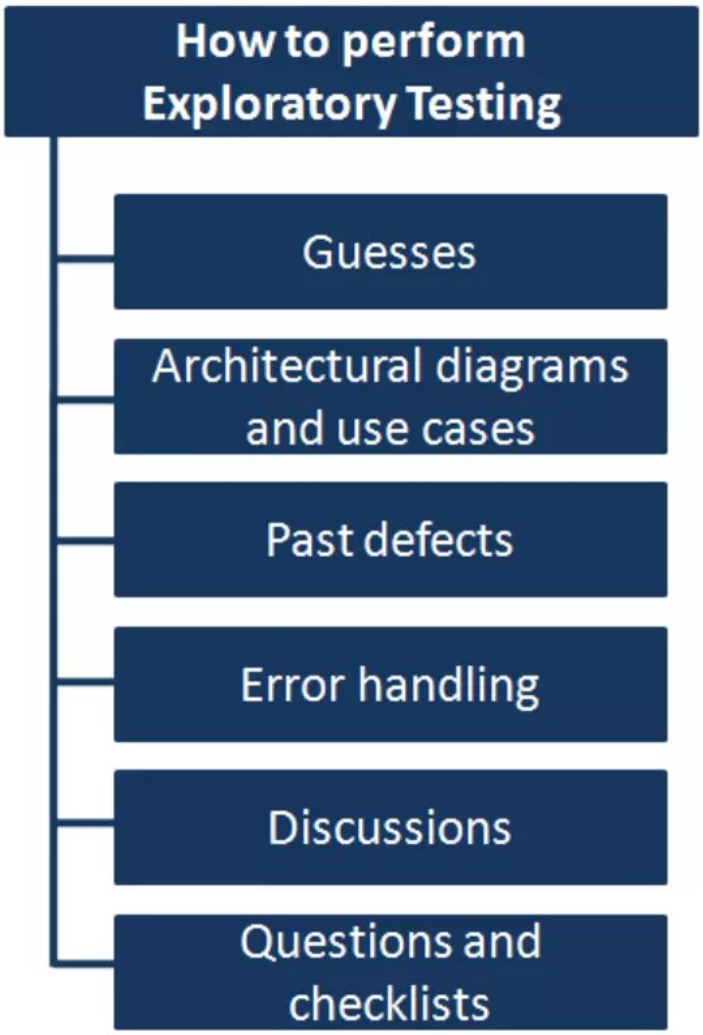
Kiểm thử thăm dò có thể thực hiện bằng cách kiểm thử với nhiều tình huống khác nhau, để có thể kiểm tra khả năng xử lý lỗi trong code tốt.

2.5. Thảo luận :

Sự hiểu biết về hệ thống thông qua quá trình giao tiếp, thảo luận và xác nhận khi làm việc, các cuộc họp, Q&A,... sẽ rất hữu ích cho việc lập kế hoạch test.

2.6. Câu hỏi và checklist :

Việc đặt ra các câu hỏi như "what, why, how, where, when,..." có thể cung cấp manh mối trong quá trình kiểm thử thăm dò.



Hình 5. Phương pháp thực hiện Exploratory testing

3. Một số lưu ý khi sử dụng Exploratory testing :

3.1. Tập trung vào mục tiêu :

- Mục tiêu của kiểm thử thăm dò là bạn đóng vai trò như một người dùng cuối để chủ động tìm kiếm và xác định lỗi.

3.2. Không tạo kịch bản, nhưng cần lập kế hoạch kiểm thử :

- Lập kế hoạch giúp làm rõ các khía cạnh cụ thể của hệ thống mà bạn muốn kiểm tra, bao gồm các yêu cầu đặc biệt hoặc mong muốn của hệ thống.

3.3. Đừng cố kiểm tra quá nhiều :

- Mục đích của kiểm thử thăm dò không phải là phạm vi đảm bảo chất lượng mà là tìm ra các khiếm khuyết mà không được phát hiện bởi các hình thức kiểm thử khác. Bản chất của nó là tập trung vào các bộ phận của hệ thống nằm ngoài các tương tác, sử dụng thông thường và ít có khả năng được kiểm thử kỹ càng.

3.4. Lưu giữ một bản ghi rõ ràng về những việc đã làm :

- Hãy ghi lại rõ ràng những task đã làm, vấn đề phát hiện ra trong một file tài liệu.

3.5. Xác định phạm vi (thời gian, quy trình) :

- Đảm bảo rằng quá trình test của bạn phải đáp ứng với thời gian và tiến độ của dự án. Cẩn thận bị cuốn sâu và tốn quá nhiều thời gian vào một chức năng nào đó, ảnh hưởng đến tiến độ chung.

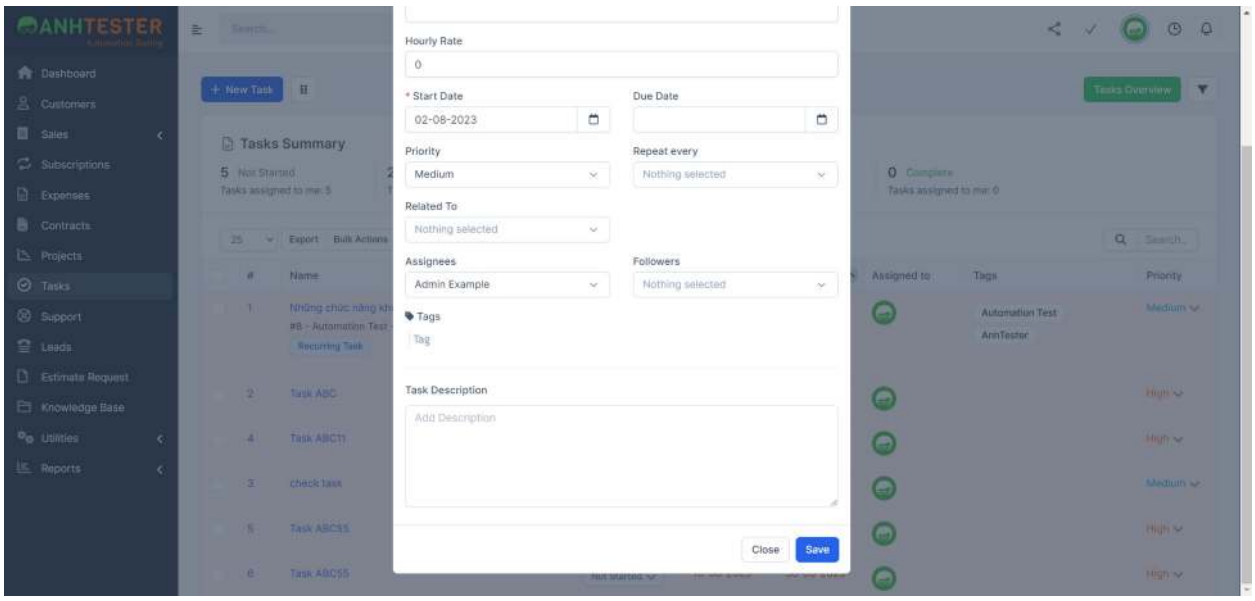
3.6. Chọn các kỹ thuật thăm dò đáp ứng với nhu cầu của dự án :

- Có các loại chính như:
    + Freestyle (adhoc testing)
    + Dựa vào scenarios
    + Dựa vào chiến lược (bảng quyết định, đồ thị nguyên nhân kết quả, đoán lỗi).
- Cần phải có sự hiểu biết về các kỹ thuật sẽ sử dụng và chia sẻ nó với các bên liên quan trong dự án như product owner, developers. Xác định phạm vi (thời gian, quy trình).

4. Một vài ví dụ khi sử dụng Exploratory testing :

4.1. Ví dụ 1 :

Bên dưới là hình ví dụ cho một form để add Task :



Các case kiểm thử khám phá có thể là:

- Click nhanh nút Add Task xem nó có hiện nhiều dialog hay không
- Sau khi mở dialog đang nhập liệu thì click chuột ra ngoài xem có bị tắt dialog hay không
- Click chuột vào nút Save nhanh nhiều lần xem nó có lưu nhiều data của form hiện tại hay không
- Dropdown list khi click tắt mở nhanh nhiều lần có bị đơ hay không
- Khi click nút Close rồi click Add Task để mở lại dialog mà làm nhanh nhiều lần xem tốc độ và phản ứng ra sao

4.2. Ví dụ 2 :

- Một khách hàng thử nghiệm ứng dụng toàn cầu sử dụng thử nghiệm khám phá trong một ứng dụng trò chơi di động iOS và Android rất phổ biến. Khi khách hàng này sẵn sàng phát hành phiên bản mới của ứng dụng của họ thường là trên cơ sở hai tháng một lần. Họ kêu gọi những người thử nghiệm khám phá từ đám động toàn cầu khám phá chức năng của ứng dụng. Những người thử nghiệm từ khắp nơi trên thế giới sẽ được đưa ra kịch bản chung từ đó kiểm tra hành vi dự kiến của phiên bản mới.

Chương V. Kết luận

1. Tổng kết :

- Trên đây là chia sẻ của mình về kiểm thử thăm dò (Exploratory testing) mà mình tìm hiểu được và thu thập được :

+ Sau khi hoàn thành đề tài về Exploratory testing, tôi đã nhận thấy rằng phương pháp này có ý nghĩa quan trọng trong việc tìm hiểu và khám phá các lỗ hổng và vấn đề trong phần mềm.

+ Qua quá trình thực hiện, tôi đã hiểu rõ hơn về cách thức tiếp cận và thực hiện exploratory testing. Phương pháp này cho phép nhóm kiểm thử tìm hiểu sâu hơn về phần mềm và phát hiện ra các lỗi mà kiểm thử kiểm soát không thể phát hiện được. Điều này giúp cải thiện chất lượng của phần mềm và đảm bảo rằng nó hoạt động đúng như mong đợi.

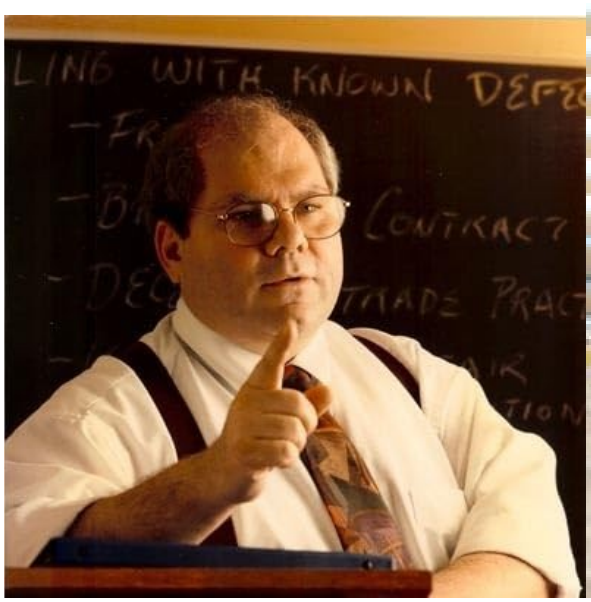
+ Một lợi ích khác của exploratory testing là khả năng linh hoạt và tùy chỉnh. Nhóm kiểm thử có thể điều chỉnh và thay đổi phương pháp theo nhu cầu và yêu cầu của dự án. Điều này giúp tăng tính linh hoạt và hiệu quả trong việc tìm ra các lỗi và vấn đề trong phần mềm.

+ Tuy nhiên, exploratory testing cũng có một số hạn chế. Phương pháp này đòi hỏi sự chuyên môn và kỹ năng từ nhóm kiểm thử. Nếu không được thực hiện đúng cách, có thể dẫn đến việc bỏ sót các lỗi quan trọng. Do đó, đội ngũ kiểm thử cần được đào tạo và có kiến thức chuyên sâu về phần mềm và quy trình kiểm thử.

+ Tổng kết lại, exploratory testing là một phương pháp quan trọng và hiệu quả trong việc tìm hiểu và khám phá các lỗ hổng và vấn đề trong phần mềm. Để đạt được hiệu quả tối đa, cần có sự chuyên môn và kỹ năng từ nhóm kiểm thử.

2. **Một số hình ảnh về Cem Kaner :**

   Cem Kaner, người đặt ra thuật ngữ này vào năm 1984 .



3. Tài liệu kham khảo :

- Lee Copeland. A Practitioner's Guide to Software Test Design. Chapter 13 .
- Cem Kaner. A Tutorial in Exploratory Testing, 2008 .
- Bach, James. Exploratory Testing Explained. v.1.3 16 April 2013
- https://www.testingexcellence.com/exploratory-testing-tips-best-practices
- Kaner, Cem,Jack Falk, and Hung Q. Nguyen (1999). Testing Computer Software. John Wiley & Sons
- Kaner, Cem,James Bach, and Bret Pettichord (2002). Lessons Learned in Software Testing: A Context-Driven Approach. John Wiley & Sons.
- https://en.wikipedia.org/wiki/Exploratory_testing
- Bach, James. "Exploratory Testing Explained." v.1.3 16 April 2003. http://www.satisfice.com/articles/et-article.pdf
- Bach, James. "Exploratory Testing and the Planning Myth." http://www.stickyminds.com/r.asp?F=DART_2359, 19 March 2001.
- https://www.google.com/
- https://www.youtube.com/