CHƯƠNG I: GIỚI THIỆU ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM

Định nghĩa phần mềm:

- Chương trình máy tính
- Các thủ tục (procedures)
- Tài liệu (documentation)
- Dữ liệu cần thiết cho sự vận hành của hệ thống

Phân loại phần mềm:

- Phần mềm hệ thống: OS như Windows, Linux, trình điều khiển thiết bị
- Phần mềm ứng dụng: MS Word, hệ thống ERP, web app, mobile app, …

Lỗi phần mềm:

- Là các phần code sai do lỗi cú pháp, logic hoặc do phân tích, thiết kế:
- Nguyên nhân gây ra lỗi phần mềm:
  + Lỗi khi định nghĩa yêu cầu
  + Quan hệ Client - Developer tồi
  + Sai phạm có chủ ý với yêu cầu phần mềm
  + Lỗi thiết kế logic
  + Lỗi lập trình
  + Không tuân thủ các hướng dẫn viết tài liệu và code
  + Thiếu sót của quá trình kiểm thử
  + Lỗi giao diện của người dùng và thủ tục
  + Lỗi tài liệu

Định nghĩa đảm bảo chất lượng phần mềm:

- **Chất lượng phần mềm là:**
  + Mức độ mà một hệ thống, thành phần hoặc một tiến trình đạt được yêu cầu đã được đặc tả
  + Mức độ mà một hệ thống, thành phần hoặc một tiến trình đạt được những nhu cầu hay mong đợi của khách hàng hoặc người sử dụng
- **Khái niệm đảm bảo chất lượng phần mềm:**
  + Một tập hợp có hệ thống các hành động nhằm đảm bảo sản phẩm phần mềm đáp ứng các yêu cầu kỹ thuật, quản lý, lịch trình và ngân sách.
  + Bao gồm cả phát triển và bảo trì phần mềm trong suốt vòng đời sản phẩm
- **Mục tiêu SQA:**
  + Đảm bảo chất lượng phần mềm và bảo trì phù hợp với các yêu cầu kĩ thuật và quản lý
  + Cải thiện hiệu quả phát triển và bảo trì phần mềm

Các yếu tố chất lượng phần mềm (McCall's Quality Factors):

- **Tiêu chí vận hành sản phẩm:**
  + Tính đúng đắn (Correctness): Đặc tả về độ chính xác, tính toàn vẹn, thời gian của outputs
  + Tính tin cậy (Reliability): Định ra tỉ lệ lỗi cho từng chức năng hoặc cả hệ thống
  + Tính hiệu quả (Efficiency): Tài nguyên phần cứng cần để thực hiện các chức năng của phần mềm
  + Tính toàn vẹn (Integrity): Bảo mật hệ thống, ngăn truy cập trái phép

+ Tính khả dụng (Usability): Tính dễ học, dễ dùng, hiệu quả

- **Tiêu chí sửa đổi sản phẩm:**
  + Tính bảo trì được (Maintainability): Mức công sức cần để tìm nguyên nhân + sửa + xác nhận đã sửa được failure (Liên quan đến cấu trúc module, kiến trúc, thiết kế và tài liệu)
  + Tính linh hoạt (Flexibility): Bảo trì, cải tiến dễ dàng
  + Tính kiểm thử được (Testability): Có lưu lại kết quả trung gian để hỗ trợ test? Có tạo file log, backup?
- **Tiêu chí chuyển giao sản phẩm:**
  + Khả năng di động (Portability): Cài trong môi trường mới (phần cứng khác, hệ điều hành khác,…) mã vẫn duy trì môi trường cũ
  + Khả năng tái sử dụng (Reusability): Có thể tái sử dụng các phần của phần mềm cho ứng dụng khác
  + Khả năng tương thích (Interoperability): Phần mềm có cần interface với các hệ thống đã có

CHƯƠNG II: TÍCH HỢP CÁC HOẠT ĐỘNG ĐBCL VÀO VÒNG ĐỜI

Các phương pháp phát triển phần mềm (Software Development Methodologies):

- **Mô hình SDLC (Waterfall – Mô hình thác nước):**
  + Gồm 7 pha: xác định yêu cầu, phân tích, thiết kế, coding, kiểm thử hệ thống, cài đặt và chuyển giao, vận hành và bảo trì
  + Đặc điểm: Tuyến tính, có trình tự, "Big Bang" (đợi đến cuối để kiểm thử)
  + Ưu điểm: Cấu trúc rõ ràng, dễ quản lý với dự án lớn
  + Nhược điểm: Không linh hoạt khi có thay đổi yêu cầu
- **Mô hình Prototyping:**
  + Ý tưởng chính: Xây dựng nguyên mẫu (Prototype) nhanh để khách hàng phản hồi
  + Ưu điểm: Phù hợp với dự án nhỏ và trung bình, tăng sự tương tác với khách hàng
  + Nhược điểm: Khó quản lý phạm vi (Scope Creep), có thể lệ thuộc vào prototype quá mức
- **Mô hình Spiral (Xoắn ốc):**
  + Đặc trưng: Tập trung vào đánh giá rủi ro, nhiều vòng lặp, phù hợp dự án phức tạp
  + Ưu điểm: Khách hàng tham gia liên tục, phát hiện rủi ro sớm
  + Nhược điểm: Tốn kém, khó thực hiện với dự án nhỏ
- **Mô hình hướng đối tượng (OO)**
  + Ý tưởng chính: Tái sử dụng thành phần phần mềm (component-based)
  + Ưu điểm: Giảm chi phí, tăng chất lượng, rút ngắn thời gian phát triển
  + Quy trình gồm: OOA, OOD, thu thập và phát triển các thành phần tái sử dụng
- **Mô hình Agile**
  + Nhấn mạnh sự lặp lại, cộng tác, phản hồi nhanh và thay đổi linh hoạt

Các mức kiểm thử (Testing Levels):

- Nguyên lý kiểm thử phần mềm:
  + Không thể chứng minh không còn lỗi
  + Kiểm thử là hoạt động dựa trên rủi ro
  + Bắt đầu càng sớm càng tốt
  + 80/20 Pareto: 20% chức năng chứa 80% lỗi
- Phân loại kiểm thử:

Mức kiểm thử	Mục tiêu	Ai thực hiện	Giai đoạn
Unit Test	Kiểm tra từng module nhỏ	Dev	Khi code xong
Integration Test	Kiểm tra các module tương tác	Dev/Test	Sau Unit Test
System Test	Kiểm thử toàn bộ hệ thống	QA	Sau tích hợp
Acceptance Test	Đảm bảo hệ thống đáp ứng yêu cầu	Khách hàng	Trước bàn giao

- Cần kiểm thử cả:
  + Yêu cầu, thiết kế, tài liệu, hệ thống thực thi
  + Các lỗi phổ biến: Ranh giới, thuật toán, giao diện, xử lý ngoại lệ

Kế hoạch đảm bảo chất lượng phần mềm (SQA Plan):

- **Chuẩn mực sử dụng:**
  + Coding, thiết kế DB, thiết kế giao diện, viết testcase
- **Các hoạt động kiểm soát chất lượng:**

+ Code walkthrough
+ Peer Review
+ Formal Review
+ Các loại test: Unit, Integration, System, Acceptance

- **Quy trình phân tích nguyên nhân (Causal Analysis):**
  + Khi xảy ra lỗi/ thành công -> Tìm được nguyên nhân gốc
- Hoạt động đánh giá và audit:
  + Audit định kỳ, cuối phase, audit theo sự cố, audit lúc bàn giao
- **Chỉ tiêu chất lượng cần đo lường (Metrics):**
  + Tỉ lệ lỗi được tiêm vào (defect injection rate)
  + Mật độ lỗi (defect density)
  + Hiệu quả loại bỏ lỗi (defect removal effciency)
  + Năng suất, sai lệch tiến độ
- **Chuẩn IEEE 730**
  + Hướng dẫn chi tiết lập SQA Plan theo chuẩn quốc tế

CHƯƠNG III: CÁC HOẠT ĐỘNG RÀ SOÁT

Mục tiêu của Rà soát (Review)

- **Mục tiêu trực tiếp (tập trung vào sản phẩm cụ thể):**
  + Phát hiện lỗi trong tài liệu phân tích và thiết kế
  + Xác định các rủi ro mới
  + Xác định sự sai lệch so với mẫu, các kiểu thủ tục và quy ước
  + Để phê chuẩn sản phẩm của phân tích hoặc thiết kế
- **Mục đích gián tiếp**
  + Nơi họp mặt không chính thức để trao đổi kiến thức chuyên môn
  + Ghi lại những lỗi phân tích và thiết kế sẽ hỗ trợ một cơ sở cho những hoạt động sửa chữa lỗi trong tương lai

Rà soát chính thức (Formal Review):

- **Các loại review điển hình:**
  + DPR - Development Plan Review: Review kế hoạch phát triển
  + SRSR - Software Requirement Specification Review: Review đặc tả yêu cầu phần mềm
  + PDR – Preliminary Design Review: Review thiết kế sơ bộ
  + DBDR - Detailed Design Review: Review thiết kế chi tiết
  + TPR – Test Plan Review: Review kế hoạch kiểm thử
  + STPR – Software Test Procedure Review: Review thủ tục kiểm thử phần mềm
  + VDR – Version Description Review: Review mô tả phiên bản
  + OMR – Operator Manual Review: Review vận hành thủ công
  + SMR – Support Manual Review: Review trợ giúp thủ công
  + TRR – Test Readiness Review: Review sự sẵn sàng kiểm thử
  + PRR – Product Release Review: Review bản phát hành sản phẩm
  + IPR – Installation Plan Review: Review kế hoạch cài dặt
- **Thành phần tham gia:**
  + Review Leader: Ngoại vi dự án, có kinh nghiệm, không phải trưởng nhóm dự án
  + Review Team: thành viên cấp cao từ phòng ban khác, 3-5 người
  + Dev Team: chuẩn bị tài liệu, thuyết trình nội dung chính

Rà soát ngang hàng (Peer Review):

- Phân biệt với Formal Review:

Đặc điểm	Formal Review	Peer Review
Người tham gia	Cấp trên, đại diện khách hàng	Thành viên cùng cấp
Mục tiêu	Phê duyệt để triển khai	Phát hiện lỗi, lệch chuẩn
Tính pháp lý	Có thể phê duyệt sản phẩm	Không có quyền phê duyệt chính thức

- Hai loại chính:
  + Inspection: Hình thức chính thức hơn, ghi lại lỗi, có báo cáo chi tiết
  + Walkthrough: Thảo luận không chính thức, chỉ đưa ra nhận xét

Triển khai hoạt động Review trong dự án:

- Cần thực hiện review ở các giai đoạn:

+ **Yêu cầu (Requirement Specification):** Kiểm tra độ đầy đủ, nhất quán, tính khả thi, tính xác minh được
+ **Thiết kế (Design Document):** Kiểm tra modularity, logic thuật toán, xử lý lỗi, cấu trúc dữ liệu
+ **SQA Plan:** Kiểm tra quy trình, timeline, tiêu chuẩn, phân công trách nhiệm

Kỹ thuật Walkthrough:

- Kỹ thuật đánh giá không chính thức (nên không có người quản lý, giám đốc dự án). Những người tham gia phải xem tài liệu trước cuộc họp (ít nhất vài ngày). Tác giả giải thích tài liệu/ sản phẩm đó cho nhóm (tác giả, điều phối viên, giám định viên, đại diện người dùng, chuyên gia bảo trì)
- Mọi người sẽ đạt câu hỏi hoặc cho ý kiến bổ sung về một số lĩnh vực để bảo đảm chất lượng kỹ thuật của tài liệu hoặc sản phẩm
- Buổi giám định có thể xảy ra vào bất kỳ lúc nào và bất kì đâu tỏng việc phát triển sản phẩm phần mềm. Mục đích chính của họp giám định chỉ là để tìm lỗi nhanh, không tìm giải pháp. Sau giám định, tác giả phải làm lại sửa mọi lỗi

Kỹ thuật Inspection:

- Kỹ thuật đánh giá chính thức. Tài liệu, sản phẩm… được những người không phải là tác giả hoặc trực tiếp liên quan (Người kiểm duyệt, tác giả, tester, thiết kế, coder) kiểm tra một cách chi tiết để phát hiện lỗi, các vi phạm tiêu chuẩn, hoặc các vấn đề khác (nếu có)
- Về cơ bản, nó được tổ chức và thực hiện chặt chẽ hơn Walkthrough. Vai trò của những người tham gia được phân định rõ ràng. Tài liệu chuẩn bị cho việc xem xét được chuẩn bị trước chu đáo.
- Quá trình duyệt thảo bắt đầu sau giai đoạn code và unit test. Sau buổi họp các lỗi tìm được sẽ được sửa lại, rồi đem ra duyệt thảo lại cho đến khi đạt tiêu chuẩn mới kết thúc quá trình này.

SQA trong tiêu chuẩn IEEE std1028:

SQA là "Tập các hoạt động có hệ thống cung cấp bằng chứng về khả năng của quy trình phần mềm tạo ra sản phẩm phần mềm khớp với việc sử dụng. Do đó hội tụ của SQA là giám sát liên tục trong toàn thể vòng đời phát triển phần mềm dể đảm bảo chất lượng của sản phẩm dược chuyển giao. Điều này yêu cầu giám sát cả quy trình và sản phẩm. Trong đảm bảo quy trình, SQA cung cấp việc quản lý với phản hồi khách quan tới tuân thủ các kế hoạch, thủ tục, chuẩn và phần tích đã được chấp thuận. Các hoạt động đảm bảo sản phẩm hội tụ vào mức độ thay đổi của chất lượng sản phẩm bên trong từng pha của vòng đời, như yêu cầu, thiết kế, viết mã và kế hoạch kiểm thử. Mục tiêu là nhận diện và khử bỏ khiếm khuyết trong toàn bộ vòng đời sớm nhất có thể được, do vậy giảm chi phí kiểm thử và bảo trì

Khái niệm Đảm bảo chất lượng phần mềm:

- SQA là tập hợp có hệ thống các hoạt động nhằm đảm bảo rằng phần mềm được phát triển dáp ứng đầy đủ các yêu cầu kỹ thuật và quản lý, bao gồm:
  + Đúng chức năng (Functional Correctness)
  + Đúng tiến độ (Schedule)
  + Trong ngân sách (Cost)
  + Dễ bảo trì, dễ mở rộng (Maintainability, Scalability)
  + Đáp ứng các yêu cầu thực tế từ người dùng cuối (Usability, Reliability, Performance)
- SQA không chỉ tập trung vào sản phẩm đầu ra, mà còn kiểm soát và cải thiện toàn bộ quá trình phát triển phần mềm, từ giai đoạn xác định yêu cầu đến bảo trì

- **SQA gồm những hoạt động:**
  + Thiết lập quy trình: Định nghĩa chuẩn, hướng dẫn, phương pháp phát triển
  + Áp dụng tiêu chuẩn: Coding standard, Design rules, Testing guideline
  + Kiểm tra chất lượng: Kiểm thử (Testing), Walkthrough, Inspections
  + Đánh giá rủi ro: Xác định, quản lý các nguy cơ tiềm ẩn trong quá trình
  + Theo dõi và đo lường; Đo defect density, defect removal rate, productivity
  + Cải tiến liên tục: Ghi nhận lỗi quá khứ để phòng tránh trong tương lai

Các mức tiêu chuẩn trong CMM (Capability Maturity Model Integration)

- **Khởi đầu:** Quy trình sản xuất phần mềm có đặc điểm tự phát, thành công chỉ dựa vào nỗ lực của các cá nhân hoặc tài năng
- **Lặp:** Các quy trình quản lý dự án cơ bản được thiết lập để kiểm soát chi phí, kế hoạch và khối lượng hoàn thành. Nguyên lý quy trình được hình thành nhằm đạt được thành công như những phần mềm tương tự
- **Xác lập:** Quy trình phần mềm cho các hoạt động quản lý cũng như sản xuất được tài liệu hóa, chuẩn hóa và tích hợp vào quy trình phần mềm chuẩn của nhà sản xuất. Các dự án sử dụng quy trình phần mềm hiệu chỉnh được phê duyệt dựa trên quy trình chuẩn của nhà sản xuất để phát triển và bảo trì sản phẩm phần mềm
- **Kiểm soát:** Thực hiện đo lường chi tiết quy trình phần mềm và chất lượng sản phẩm.Cả quy trình sản xuất và sản phẩm phần mềm được kiểm soát theo định lượng
- **Tối ưu:** Quy trình liên tục được cải tiến dựa trên những ý kiến phản hồi từ việc sử dụng quy trình, thí điểm những ý tưởng quản lý và công nghệ mới

Mục tiêu của SQA – Hoạt động chính:

- **Mục tiêu của SQA trong phát triển phần mềm**
  + Đảm bảo một mức độ tin cậy chấp nhận được là phần mềm sẽ tuân thủ các yêu cầu kĩ thuật về chức năng
  + Đảm bảo một mức độ tin cậy chấp nhận được là phần mềm sẽ tuân thủ các yêu cầu quản lý về thời gian, tài chính
  + Khởi đầu và quản lý các hoạt động để phát triển phần mềm và các hoạt động SQA được cải thiện và đạt hiệu quả cao hơn
- **Mục tieu của SQA trong bảo trì phần mềm:**
  + Đảm bảo một mức độ tin cậy chấp nhận được là các hoạt động bảo trì phần mềm sẽ tuân thủ các yêu cầu kĩ thuật về chức năng
  + Đảm bảo một mức độ tin cậy chấp nhận được là các hoạt động bảo trì phần mềm sẽ tuân thủ các yêu cầu quản lý thời gian và tài chính
  + Khởi đầu và quản lý các hoạt động để bảo trì phần mềm và các hoạt động SQA được cải thiện và đạt hiệu quả cao hơn
- **7 hoạt động chính:**
  + Áp dụng công nghệ kĩ thuật hiệu quả (phương pháp, công cụ)
  + Tiến hành rà soát kĩ thuật chính thức
  + Thực hiện kiểm thử nhiều tầng
  + Tuần theo các chuẩn phát triển
  + Kiểm soát tài liệu PM và thay đổi của chúng

+ Thực hiện do lường
+ Báo cáo và quản lý các bảo cáo

Khảo sát nhu cầu SQA:

- Kiểm kê các chính sách SQA: chính sách, thủ tục, chuẩn nào đã có trong các pha phát triển
- Đánh giá vai trò của kỹ nghệ phần mềm, đảm bảo chất lượng trong tổ chức hiện tại có quyền lực đến đâu
- Đánh giá mối quan hệ SQA: Giao diện chức năng giữa SQA với các đơn vị khác như thế nào? Với người thực hiện rà soát kỹ thuật chính thức, quản lý cấu hình và thử nghiệm Khởi đầu và quản lý các hoạt động để phát triển phần mềm và các hoạt động SQA được cải thiện và đạt hiệu quả cao hơn

CHƯƠNG IV: KIỂM THỬ PHẦN MỀM

Khái niệm ca kiểm thử - Mục đích thiết kế ca kiểm thử:

- Một ca kiểm thử (testcase) trong CNPM là một tập hợp các diều kiện hay các biến để theo đó một thử nghiệm sẽ xác định xem một ứng dụng hoặc hệ thống phần mềm đang làm việc một cách chính xác hay không
- Mục đích:
  + Muốn tìm ra được nhiều sai nhất với nỗ lực và thời gian là nhỏ nhất
  + Chứng minh dược sự tồn tại của lỗi
  + Không chứng minh được sự không có lỗi

CHƯƠNG V: KIỂM THỬ HỘP ĐEN – KIỂM THỬ HỘP TRẮNG:

Khái niệm kiểm thử hộp trắng – Các đặc trưng:

- Là hình thức kiểm thử mà tester biết được các cấu trúc bên trong của chương trình (mã nguồn, xử lý dữ liệu,…). Việc kiểm thử dược dựa trên các phân tích về cấu trúc bên trong của thành phần/ hệ thống
- Kiểm tra mã nguồn các chi tiết thủ tục (thuật toán), các con đường logic (luồng điều khiển), các trạng thái của chương trình (dữ liệu)
- Đặc trưng:
  + Kiểm thử hộp trắng dựa vào thuật giải cụ thể, vào cấu trúc dữ liệu bên trong của đơn vị phần mềm cần kiểm thử để xác định đơn vị phần mềm đó có thực hiện đúng không
  + Người kiểm thử hộp trắng phải có kỹ năng, kiến thức nhất định để có thể hiểu chi tiết về đoạn code cần kiểm thử
  + Thường tốn rất nhiều thời gian và công sức nếu mức độ kiểm thử được nâng lên ở cấp kiểm thử tích hợp hay kiểm thử hệ thống
  + Do đó kỹ thuật này chủ yếu được dùng để kiểm thử đơn vị. Trong OOP, kiểm thử đơn vị là kiểm thử từng tác vụ của 1 class chức năng nào đó
  + Có 2 hoạt động kiểm thử hộp trắng: Kiểm thử luồng diều khiển và kiểm thử dòng dữ liệu

Khải niệm kiểm hộp đen – Các đặc trưng:

- Hình thức kiểm thử mà tester không cần biết dến cách thức hoạt động, mã nguồn, xử lý dữ liệu bên trong 1 thành phần/ hệ thống
- Công việc cần làm là nhập dữ liệu đầu vào (input) và kiểm tra kết quả trả về có đúng như mong muốn hay không
- Đặc trưng:
  + Cơ sở: đặc tả, các điều kiện vào/ ra và cấu trúc dữ liệu
  + Nhằm thuyết minh các chức năng phần mềm đủ và vận hành đúng
  + Thực hiện các phép thử qua giao diện
  + Thường phát hiện các lỗi đặc tả yêu cầu, thiết kế
  + Dễ dàng thực hiện
  + Chi phí thấp
  + Được sử dụng để kiểm thử phần mềm tại mức: modul, tích hợp, hàm, hệ thống
  + Đơn giản hóa kiểm thử tại các mức độ được đánh giá là khó kiểm thử
  + Khó đánh giá còn bộ giá trị nào chưa được kiểm thử hay không
  + Ít chú ý tới cấu trúc logic nội tại của nó

Quy trình kiểm thử phần mềm (Software Testing Process):

- Test Planning: Xác định mục tiêu, chiến lược, tài nguyên, lịch kiểm thử
- Monitoring + Control: So sánh tiến độ thực tế vs kế hoạch, đưa ra điều chỉnh nếu lệch
- Test Analysis: Xác định cái gì cần kiểm thử (test condition), phân tích yêu cầu, thiết kế
- Test Design: Thiết kế test case, xác định dữ liệu kiểm thử và môi trường
- Test Implementation: Chuẩn bị testware, mã hóa testcase, thiết lập môi trường test
- Test Execution: Thực thi testcase, ghi nhận kết quả, phân tích lỗi
- Test Completion: Tổng hợp, đánh giá, lưu trữ kết quả, rút kinh nghiệm

Kỹ thuật kiểm thử hộp đen:

- **Equivalence Partitioning (Phân vùng tương đương):**
  + Chia miền giá trị đầu vào thành các nhóm có hành vi giống nhau
  + Mỗi nhóm gọi là 1 partition (hoặc equivalence class)
  + Chỉ cần chọn 1 giá trị đại diện cho mỗi nhóm để kiểm thử -> giảm số lượng testcase
- **Boundary Value Analysis (Phân tích giá trị biên)**
  + Mở rộng từ kỹ thuật phân vùng
  + Lỗi thường xảy ra ở biên của phân vùng (boundary)
  + Test tại giá trị min, max và xung quanh nó (VD: min-1,min,min+1;max-1,max,max+1)
- **Decision Table Testing (Bảng quyết định)**
  + Dùng để kiểm thử các luật nghiệp vụ phức tạp, dựa trên các tổ hợp điều kiện
  + Bảng quyết định gồm:
    Hàng: Điều kiện và hành động
    Cột: Các luật quyết định (rule) -> mỗi cột tương ứng có 1 testcase
- **State Transition Testing (Kiểm thử chuyển trạng thái):**
  + Dùng khi hành vi hệ thống phụ thuộc vào trạng thái hiện tại
  + Dựa trên sơ đồ/ trạng thái (state diagram) và bảng chuyển đổi
  + Xây dựng testcase để bao phủ:
    Tất cả các trạng thái
    Tất cả các chuyển trạng thái (transitions)
    Các chuỗi chuyển trạng thái
- **Pairwise Testing (Kiểm thử kết hợp từng cặp):**
  + Phân tích các kết hợp cặp giá trị đầu vào vì phần lớn lỗi xảy ra khi 2 giá trị tương tác với nhau.
  + Giảm số lượng test case từ hàng trăm → chỉ còn vài chục mà vẫn bao phủ logic chính
  + Sử dụng tool (VD: Microsoft PICT) để tạo bộ test tối ưu theo các cặp.

Kỹ thuật kiểm thử hộp trắng:

- **Control Flow Testing (Kiểm thử luồng điều khiển):**
  + Dựa vào sơ đồ điều khiển (Control Flow Graph – CFG) để xác định đường đi (path) qua đoạn mã
- Khái niệm quan trọng:
  + Execution Path: Đường thực thi từ điểm bắt đầu -> Kết thúc trong mã nguồn
  + Coverage (bao phủ): % thành phần được kiểm tra so với toàn bộ mã
- **Data Flow Testing (Kiểm thử luồng dữ liệu):**
  + Mục tiêu:
    Kiểm tra các quá trình khởi tạo, sử dụng, và hủy biến trong suốt vòng đời
    Xác định các lỗi như: Sử dụng biến chưa được gán, gán biến và không sử dụng, hủy biến sai chỗ

CHƯƠNG VI: CÁC CÔNG CỤ SQA (SQA TOOLS):

Có những loại công cụ tự động nào giúp trợ giúp kiểm thử, mô tả nội dung của mỗi loại

- Công cụ kiểm thử tự động (Testing Tools) được chia thành những nhóm chính:
  + **Design:** Các công cụ giúp quyết định test cần gì để được thực thi. Test data và testcase sinh ra (generators). Testing tools giúp tạo ra các testcase, hoặc số đầu vào test tối thiểu (là 1 phần của testcase). Có khoảng 15 tools
  + **GUI (Graphical User Interface):** Các công cụ này tự động thực thi các test cho products với giao diện người dùng Graphic. Công cụ tự động kiểm tra Client/Server, bao gồm cả load testers. GUI Testing là quá trình kiểm thử giao diện người dùng của ứng dụng và phát hiện các chức năng chính xác của ứng dụng
  + **Load and Performance:**
  Load testing là quá trình đặt nhu cầu về hệ thống hoặc thiết bị và đo sự phản ứng của nó. Load testing được thực hiện để xác định hành vi của hệ thống theo 2 điều kiện tải bình thường và được mong đợi. Nó giúp để xác định khả năng hoạt động tối đa của ứng dụng cũng như bất kỳ vướng mắc và xác định yếu tố gây ra suy thoái
  Performance testing được thực hiện để xác định một hệ thống có đáp ứng được yêu cầu và ổn định trong khối công việc cụ thể. Nó cũng có thể phục vụ việc kiểm tra, đo lường, xác minh chất lượng các thuộc tính của hệ thống, chẳng hạn như khả năng mở rộng, độ tin cậy và sử dụng tài nguyên
  + **Manangement:** Được sử dụng để lưu trữ thông tin quá trình kiểm thử làm việc như thế nào, kế hoạch hoạt động kiểm thử và báo cáo tình hình hoạt động của SQA
  + **Implementation:** Dụng cụ khác giúp thực hiện bài kiểm tra. Ví dụ, công cụ tự động tạo ra thói quen khi còn sơ khai ở đây, cũng như các công cụ mà cố gắng để làm cho thất bại rõ ràng hơn (máy phát điện khẳng định)
  + **Evaluation (Sự đánh giá):** Công cụ giúp đánh giá chất lượng các bài kiểm tra. Các công cụ bảo hiểm mã đi đây
  + **Static Analysis (Phân tích tĩnh):** Công cụ phân tích các chương trình mà không cần chạy chúng. Số liệu công cụ rơi vào thể loại này
  + Defect Tracking, Website và Miscellaneous (Hỗn hợp)

Đồ thị luồng điều khiển gồm những yếu tố nào? Dùng để làm gì

- Đồ thị dòng điều khiển là 1 đồ thị có hướng gồm các đỉnh tương ứng với các câu lệnh/ nhóm câu lệnh và các cạnh là các dòng điều khiển giữa các câu lệnh/ nhóm câu lệnh
- Nhưng yếu tố trong đồ thị luồng điều khiển:
  + Điều bắt đầu của đơn vị chương trình
  + Khối xử lý chứa các câu lệnh được khai báo hoặc tính toán
  + Điểm quyết định ứng với các câu lệnh điều kiện trong các khối lệnh rẽ nhánh hoặc lặp
  + Điểm nối ứng với các câu lệnh ngay sau các lệnh rẽ nhánh
  + Điểm kết thúc ứng với điểm kết thúc của đơn vị chương trình
- Mục tiêu của phương pháp kiểm thử luồng điều khiển là đảm bảo mọi đường thi hành của đơn vị phần mềm cần kiểm thử đều chạy đúng. Rất tiếc trong thực tế, công sức và thời gian để đạt mục tiêu trên là rất lớn, ngay cả trên đơn vị phần mềm nhỏ

Đồ thị luồng dữ liệu gồm những yếu tố nào? Dùng để làm gì

- Đồ thị dòng dữ liệu của một chương trình/ đơn vị chương trình là một đồ thị có hướng
  G = <N;E> với:
  + N là tập các đỉnh tương ứng với các câu lệnh def hoặc c-use của các biến được sử dụng trong đơn vị chương trình. Đồ thị G có 2 đỉnh đặc biệt là đỉnh bắt đầu (tương ứng với lệnh def của biến tham số) và đỉnh kết thúc đơn vị chương trình
  + E là tập các cạnh tương ứng với các câu lệnh p-use của các biến
- Lý do cần kiểm thử dòng dữ liệu:
  + Cần chắc chắn biến được gán đúng giá trị, tức là ta phải xác định được một đường đi của biến từ 1 điểm bắt đầu nơi nó được định nghĩa đến điểm mà biến đó được sử dụng
  + Ngay cả khi gán đúng giá trị cho biến thì các giá trị được sinh ra chưa chắc đã chính xác do tính toán hoặc các biểu thức diều kiện sai (biến được sử dụng sai)

Chiến lược kiểm thử phân nhánh (Branch) là gì?

- Kiểm thử phân nhánh là chiến lược kiểm thử từng điều kiện chương trình
- Kiểm thử nhánh: Với mỗi điều kiện kết hợp C thì các nhánh true và false của C và mỗi điều kiện đơn trong C phải được kiểm thử ít nhất 1 lần
- Yêu cầu: Không chỉ phát hiện sai trong điều kiện đó mà còn phát hiện các sai khác trong chương trình

Kiểm thử đơn vị là gi? Hoạt động kiểm thử đơn vị gồm những nội dung gì? Liên quan nhân tố nào?

- Kiểm thử đơn vị là quá trình kiểm thử các thành phần riêng biệt của hệ thống. Đây là quá trình kiểm thử khiếm khuyết vì vậy mục tiêu của nó là tìm ra lỗi trong các thành phần
- Kiểm thử đơn vị có các nội dung:
  + Kiểm thử giao diện
  + Khám nghiệm cấu trúc dữ liệu cục bộ
  + Kiểm thử với các điều kiện biên
  + Các dường độc lập
  + Các đường xử lý sai
- Các nhân tố liên quan:
  + Tham số
  + Vào ra
  + Dữ liệu cục bộ

Kiểm thử tích hợp là gì? Thực hiện khi nào?

- Kiểm thử tích hợp là một kĩ thuật có tính hệ thống để xây dựng cấu trúc chương trình ngay khi đang tiến hành kiểm thử để phát hiện sai liên kết với giao diện. Kiểm thử tích hợp nhằm nhận được một phần hay toàn bộ hệ thống như mong đợi
- Kiểm thử tích hợp được thực hiện sau khi đã thực hiện kiểm thử đơn vị và ghép nối các đơn vị/ thành phần phần mềm
- Mục đích: Tận dụng các modul đã kiểm thử đơn vị và xây dựng chương trình sao cho nó đảm bảo theo thiết kế
- Phải kiểm thử tích hợp vì:
  + Dữ liệu có thể bị mất khi đi qua một giao diện
  + Một module có thể có một hiệu ứng bất lợi vô tình lên các module khác

+ Các chức năng phụ khi kết hợp lại có thể không sinh ra chức năng chính mong muốn
+ Các điều không chính xác riêng rẽ có thể bị phóng đại đến mức không chấp nhận được
+ Các cấu trúc dữ liệu toàn cục có thể để lộ ra các vấn đề

- Ưu điểm:
  + Dễ dàng tìm ra các lỗi vào ngay giai doạn đầu
  + Dễ dàng khoanh vùng các lỗi (tích hợp n module, n+1 module)
  + Giảm việc sử dụng các stub Driver

Nội dung chính của kiểm thử hệ thống:

- Hệ thống dựa vào máy tính do nhiều bên xây dựng, người phát triển phần mềm chỉ là một
- Việc kiểm thử hệ thống để có nguy cơ "đổ lỗi cho nhau"
- Người phát triển phần mềm cần đoán trước các vấn đề giao diện có thể xảy ra
- Phát hiện các thiết kế đường xử lý sai thông qua kiểm thử tất cả các thông tin đến từ các phần tử khác của hệ thống
- Tiến hành một loại kiểm thử mô phỏng các dữ liệu xấu hoặc cái sai tiềm tàng khasctaij giao diện phần mềm
- Báo cáo các kết quả kiểm thử để làm chứng cớ phòng ngừa đổ lỗi cho nhau
- Những người tham gia vào trong việc hoạch định và thiết kế các kiểm thử hệ thống để đảm bảo rằng phần mềm được kiểm thử dầy đủ
- Việc kiểm thử hệ thống thực tế là một loại các bước kiểm thử khác nhau có mục đích chính là thử đầy đủ hệ thống dựa trên máy tính

Khi nào nên dùng Test Tool? Ưu nhược điểm

Test Tool được dùng khi:

- Không đủ tài nguyên: Khi số lượng testcase quá nhiều mà kiểm thử viên không thể hoàn tất trong thời gian cụ thể
- Kiểm tra hồi quy: Nâng cấp phần mềm, kiểm tra lại các tính năng đã chạy tốt và những tính năng đã sửa. Tuy nhiên, việc này khó đảm bảo về mặt thời gian
- Kiểm tra khả năng vận hành phầm mềm trong môi trường đặc biệt:
  + Đo tốc độ trung bình xử lý 1 yêu cầu của Web Server
  + Xác định số yêu cầu tối đa được xử lý bởi Web Server
  + Xác định cấu hình máy thấp nhất mà PM vẫn có thể hoạt động tốt

Ưu điểm

- Giảm bớt công sức và thời gian thực hiện quá trình kiểm thử
- Tăng độ tin cậy
- Giảm sự nhàm chán cho con người
- Rèn luyện kĩ năng lập trình cho tester
- Giảm chi phí cho tổng quá trình kiểm thử

Nhược điểm:

- KTPM khoogn cần can thiệp của tester
- Giả lập tình huống khó có thể thực hiện bằng tay
- Mất chi phí tạo ra và bảo trì các script để thực hiện kiểm tra tự động

- Khó bảo trì, khó mở rộng các script
- Đòi hỏi tester phải có kĩ năng tạo script kiểm tra tự động
- Không áp dụng được trong việc tìm lỗi mới của PM

Quy trình kiểm thử phần mềm:

- **Phân tích yêu cầu:** Kiểm thử thường sẽ bắt đầu lấy các yêu cầu trong các giai đoạn của vòng đời phần mềm. Trong giai đoạn thiết kế, các tester làm việc với các nhà phát triển để xác định những khía cạnh của một thiết kế được kiểm chứng và những thông số được kiểm tra
- **Lập kế hoạch kiểm thử:** Chiến lược kiểm thử, kế hoạch kiểm thử, kiểm thử sáng tạo,… Và có một kế hoạch là cần thiết vì nhiều hoạt động sẽ được thực hiện trong thời gian kiểm thử
- **Kiểm thử phát triển:** Các quy trình kiểm thử, các kịch bản, testcase, các dữ liệu được sử dụng trong kiểm thử phần mềm
- **Kiểm thử thực hiện**: Dựa trên các kế hoạch, các văn bản kiểm thử và các báo cáo bất kỳ lỗi nào tìm thấy cho nhóm phát triển
- **Kiểm thử báo cáo**: Sau khi hoàn tất kiểm thử, các tester tạo ra các số liệu và báo cáo cuối cùng về nỗ lực kiểm thử của họ và có sẵn sàng phát hành phần mềm hay không
- Phân tích kết quả kiểm thử hoặc phân tích thiếu sót được thực hiện bởi đội ngũ phát triển kết hợp với khách hàng để đưa ra quyết định xem những thiếu sót gì cần phải dược chuyển giao, cố định và tử bỏ (tức là tìm ra được phần mềm hoạt động chính xác), hoặc giải quyết sau
- **Test lại khiếm khuyết:** Khi một khiêm khuyết đã được xử lý bởi đội ngũ phát triển, nó phải được kiểm tra lại bởi nhóm kiểm thử
- **Kiểm thử hồi quy:** Người ta thường xây dựng 1 chương trình kiểm thử nhỏ là tập hợp của các bài test cho mỗi tích hợp mới, sửa chữa hoặc cố định phần mềm, để đảm bảo rằng những cung cấp mới nhất đã không phá hủy bất cứ điều gì và toàn bộ phần mềm vẫn còn hoạt động chính xác

Test Plan là gi? Gồm nội dung gì?

- Là một tài liệu mô tả các mục tiêu, phạm vi, phương pháp tiếp cận, và tập trung vào nỗ lực kiểm thử phần mềm. Quá trình chuẩn bị test plan là một cách hữu ích để suy nghĩ tới những nỗ lực cần thiết dể xác nhận khả năng chấp nhận một sản phẩm phần mềm. Các tài liệu đã hoàn thành sẽ giúp mọi người bên ngoài nhóm test hiểu được tại sao và như thế nào chấp nhận sản phẩm. Nó cần phải hoàn hảo để dùng được nhưng không đủ hoàn hảo vì không ai ngoài nhóm test sẽ đọc nó

Nội dung:

Chiến lược kiểm tra

- Chiến lược kiểm tra đưa ra phương pháp tiếp cận để kiểm tra mục tiêu
- Chiến lược kiểm tra bao gồm các kỹ thuật được áp dụng và điều kiện để biết khi nào việc kiểm tra hoàn thành
  + Mô tả các kiểu kiểm tra dùng trong dự án
  + Có thể liệt kê với mỗi kiểu kiểm tra tương ứng kiểm trac ho chức năng nào
  + Việc kiểm tra có thể dừng khi nào

Các kiểu kiểm tra

- Môi kiểu kiểm tra phải bao gồm các điều kiện:

+ Kỹ thuật: Mô tả việc kiểm tra như thế nào, những gì sẽ được kiểm tra, các hoạt động chính được thực hiện trong quá trình kiểm tra và các phương pháp đánh giá kết quả
+ Điều kiện hoàn thành: Xác định chất lượng chương trình được chấp thuận, thời điểm kiểm tra hoàn tất
+ Các vấn đề đặc biệt liên quan: Vấn đề gây ảnh hưởng đến việc kiểm tra

- Kiểm tra chức năng
- Kiểm tra hiệu suất
- Kiểm tra bảo mật và kiểm soát truy cập
- Kiểm tra hồi quy

Môi trường kiểm tra

Tại sao phải kiểm thử? Mục tiêu kiểm thử?

- **Lý do:**
  + Kiểm thử là yếu tố quyết định tổng SQA và khâu điển hình của rà soát đặc tả thiết kế và lập mã
  + Muốn nhìn thấy phần mềm như là 1 phần tử của hệ thống hoạt động (xem sản phẩm)
  + Hạn chế chi phí phải trả cho các thất bại do lỗi gây ra sau này (hạu quả)
  + Có kế hoạch tốt nâng cao chất lượng cho suốt quá trình phát triển (giải pháp)
- **Tầm quan trọng của kiểm thử:**
  + Chi phí của kiểm thử chiếm 40% tổng công sức phát triển và trên 30% tổng thời gian phát triển
  + Với phần mềm ảnh hưởng đến sinh mạng chi phí có thể gấp 3 đến 5 lần tổng chi phí khác cộng lại
- **Mục tiêu trước mắt:**
  + Cố gắng tạo ra các ca kiểm thử để chỉ ra lỗi của phần mềm với chi phí thấp nhất. Một ca kiểm thử thắng lợi là phải làm lộ ra khiếm khuyết, đồng thời mang lại các lợi ích phụ
- **Mục tiêu cuối cùng:** Có 1 chương trình tốt, chi phí ít

1. Checklist GUI:

No.	Item
1	Is the user interface intuitive? **Giao diện người dùng có trực quan không?**
2	Does the system allow for easy registration of students? **Hệ thống có cho phép sinh viên đăng ký dễ dàng không?**
3	Can administrators easily access and manage student registration data? **Người quản lý có thể dễ dàng truy cập và quản lý dữ liệu đăng ký của sinh viên không?**
4	Is the system efficient in handling a large volume of registrations? **Hệ thống có hiệu quả trong việc xử lý khối lượng đăng ký lớn không?**
5	Does the system provide accurate and timely registration information? **Hệ thống có cung cấp thông tin đăng ký chính xác và kịp thời không?**
6	Is the registration process streamlined for both students and administrators? **Quá trình đăng ký có được sắp xếp hợp lý cho cả sinh viên và người quản lý không?**
7	Are there clear error messages for any registration issues? **Có thông báo lỗi rõ ràng nào cho bất kỳ vấn đề** đăng ký nào không?
8	Does the system have adequate security measures to protect student data? **Hệ thống có các biện pháp bảo mật đầy đủ để bảo vệ dữ liệu của sinh viên không?**
9	Can the system generate reports related to registration statistics? **Hệ thống có thể tạo báo cáo liên quan đến số liệu thống kê đăng ký không?**
10	Is the system user-friendly for both novice and experienced users? **Hệ thống này có thân thiện với người dùng mới và người dùng có kinh nghiệm không?**

2. Checklist Test Case for Function Test:

Item No.	Item
General	Is the user interface true to the prototype? **Giao diện người dùng có giống với nguyên mẫu?**
	Are menu lists and screens consistent? **Danh sách menu và màn hình có nhất quán k?**
	Do html links work? **Liên kết HTML có hoạt động không?**
	Are all windows accessible from the toolbar? **Có thể truy cập tất cả các cửa sổ từ thanh công cụ không?**
	Are all web pages/windows accessible from the menu? **Có thể truy cập tất cả các trang web/cửa sổ từ menu không?**
	Are the screens called from the button displayed correctly? **Màn hình được gọi từ nút có hiển thị đúng không?**
	Labels, textboxes, combo boxes, etc. have the correct font, font size, and color upon request? **Label, textbox, combo boxes, v.v. có phông chữ, cỡ chữ và màu sắc chính xác theo yêu cầu không?**
	Are the alignment, width, and spacing consistent and correct as required? **Sự căn chỉnh, chiều rộng và khoảng cách có nhất quán và chính xác theo yêu cầu không?**
	Left-aligned font style data? **Dữ liệu kiểu phông chữ căn trái?**
	Right-aligned numeric data? **Dữ liệu số căn phải?**
	Are the forms well laid out and easy to use? **Các forms có được trình bày rõ ràng và dễ sử dụng không?**
	If keyboard shortcuts are used, the assigned keyboard shortcuts are active? **Nếu sử dụng phím tắt, các phím tắt được chỉ định có được kích hoạt không?**
	Does the screen have NO spelling, sentence structure, or grammar errors? **Màn hình KHÔNG có lỗi chính tả, lỗi cấu trúc câu hoặc lỗi ngữ pháp phải không?**
	If abbreviations are used, are they consistent throughout interface? Can users understand it? **Nếu sử dụng chữ viết tắt, chúng có nhất quán trong toàn bộ giao diện**

	không? Người dùng có thể hiểu được không?
	Is the format of numbers, dates, and times consistent? **Định dạng số, ngày tháng và thời gian có nhất quán không?**
	Does the description show when moving the mouse over the tooltip? **Mô tả có hiển thị khi di chuyển chuột qua chú giải công cụ không?**
	If there is a tooltip, is it meaningful or useful? **Nếu có tooltip thì nó có ý nghĩa hoặc hữu ích không?**
	Is there a button or check box that is considered default? **Có nút hoặc hộp kiểm nào được coi là mặc định không?**

	The cursor moves in order: from left to right, from top to bottom under pressing Tab repeatedly? **Con trỏ di chuyển theo thứ tự: từ trái sang phải, từ trên xuống dưới khi nhấn Tab nhiều lần?**
	The cursor moves in order: from bottom to top, from right to left when pressing Shift-Tab? **Con trỏ di chuyển theo thứ tự: từ dưới lên trên, từ phải sang trái khi nhấn Shift-Tab?**
	If the mouse does not focus on any button, the function has been performed main button when pressing Enter? **Nếu chuột không tập trung vào bất kỳ nút nào thì chức năng của nút chính đã được thực hiện khi nhấn Enter?**
	If you are focusing on a button, the button's function has been performed. Press Enter yet? **Nếu bạn đang tập trung vào một nút, chức năng của nút đó đã được thực hiện. Nhấn Enter chưa?**
	The screen zooms out and zooms in accordingly and the interface does not break when press Ctrl - and Ctrl +? **Màn hình thu nhỏ và phóng to tương ứng và giao diện không bị hỏng khi nhấn Ctrl - và Ctrl +?**
	Are required fields marked with *? **Các trường bắt buộc có được đánh dấu * không?**
	The record information displayed on the interface is correct with the record information saved in Database? **Thông tin bản ghi hiển thị trên giao diện có đúng với thông tin bản ghi được lưu trong Database không?**
	Check the language change, the changed language has the correct content **Kiểm tra thay đổi ngôn ngữ, ngôn ngữ đã thay đổi có nội dung đúng**

	Fields that do NOT allow data to be entered have the same color as required request? **Các trường KHÔNG cho phép nhập dữ liệu có cùng màu với yêu cầu bắt buộc không?**
Validate the data fields **Xác thực các trường dữ liệu**	**Đối với các kiểu dữ liệu như văn bản, chuỗi, v.v.:**
	Check if the max length meets the requirements? **Kiểm tra xem độ dài tối đa có đáp ứng yêu cầu không?**
	If the data field is case sensitive, is the input data satisfied? **Nếu trường dữ liệu phân biệt chữ hoa chữ thường thì dữ liệu đầu vào có thỏa mãn không?**
	If the data field is NOT case sensitive, is the input data satisfied? **Nếu trường dữ liệu KHÔNG phân biệt chữ hoa chữ thường thì dữ liệu đầu vào có thỏa mãn không?**
	If the data field allows null, is it satisfied? **Nếu trường dữ liệu cho phép giá trị null thì có thỏa mãn không?**
	If the data field does NOT allow nulls, is it satisfied? **Nếu trường dữ liệu KHÔNG cho phép giá trị null thì có thỏa mãn không?**
	If the data field allows special characters, the data is entered. satisfied? **Nếu trường dữ liệu cho phép ký tự đặc biệt thì dữ liệu được nhập vào có thỏa mãn không?**
	If the data field does NOT allow special characters, is the input data satisfied? **Nếu trường dữ liệu KHÔNG cho phép ký tự đặc biệt, dữ liệu đầu vào có thỏa mãn không?**
	Đối với các kiểu dữ liệu là số nguyên, float, double, v.v.:
	Check if the max length meets the requirements? **Kiểm tra xem độ dài tối đa có đáp ứng yêu cầu không?**
	Check whether the boundary values meet the requirements? **Kiểm tra xem giá trị ranh giới có đáp ứng yêu cầu không?**

	If the data field allows special characters, is the input data satisfied? **Nếu trường dữ liệu cho phép các ký tự đặc biệt, dữ liệu đầu vào có đáp ứng được không?**
	If the data field does NOT allow special characters, is the input data satisfied?

	Nếu trường dữ liệu KHÔNG cho phép ký tự đặc biệt, dữ liệu đầu vào có thỏa mãn không?
	If the data field allows null, is it satisfied? **Nếu trường dữ liệu cho phép giá trị null thì có thỏa mãn không?**
	If the data field does NOT allow nulls, is it satisfied? **Nếu trường dữ liệu KHÔNG cho phép giá trị null thì có thỏa mãn không?**
	If the data field allows text characters to be entered, is the input data satisfied? **Nếu trường dữ liệu cho phép nhập ký tự văn bản thì dữ liệu đầu vào có thỏa mãn không?**
	If the data field does NOT allow entering text data, is the input data satisfied? **Nếu trường dữ liệu KHÔNG cho phép nhập dữ liệu văn bản, dữ liệu đầu vào có thỏa mãn không?**
	Check division by zero error? **Kiểm tra lỗi chia cho số không?**
	Đối với kiểu dữ liệu thời gian và ngày tháng:
	Check if the max length meets the requirements? **Kiểm tra xem độ dài tối đa có đáp ứng yêu cầu không?**
	Check whether the day, month, year, hour, minute, second are valid? **Kiểm tra xem ngày, tháng, năm, giờ, phút, giây có hợp lệ không?**
	If the data field allows special characters, is the input data satisfied? **Nếu trường dữ liệu cho phép các ký tự đặc biệt, dữ liệu đầu vào có đáp ứng được không?**
	If the data field does NOT allow special characters, is the input data satisfied **Nếu trường dữ liệu KHÔNG cho phép các ký tự đặc biệt, dữ liệu đầu vào có thỏa mãn không?**
	If the data field allows null, is it satisfied? **Nếu trường dữ liệu cho phép giá trị null thì có thỏa mãn không?**
	If the data field does NOT allow nulls, is it satisfied? **Nếu trường dữ liệu KHÔNG cho phép giá trị null thì có thỏa mãn không?**
	If the data field allows text characters to be entered, is the input data satisfied? **Nếu trường dữ liệu cho phép nhập ký tự văn bản thì dữ liệu đầu vào có thỏa mãn không?**
	Does it allow users to click on dates? **Nó có cho phép người dùng nhấp vào ngày không?**
	Is it allowed for users to enter dates?

	Người dùng có được phép nhập ngày tháng không?
	If the data field does NOT allow entering text data, is the input data satisfied? **Nếu trường dữ liệu KHÔNG cho phép nhập dữ liệu văn bản, dữ liệu đầu vào có thỏa mãn không?**
	Check if the format meets the requirements? **Kiểm tra xem định dạng có đáp ứng yêu cầu không?**
	Đối với kiểu dữ liệu tệp:
	Ensure minimum file limit? **Đảm bảo giới hạn tệp tối thiểu?**
	Is the maximum file limit guaranteed? **Giới hạn tệp tối đa có được đảm bảo không?**
	Is it allowed to upload multiple files at once? **Tôi có thể tải lên nhiều tệp cùng một lúc không?**
	Are nulls allowed? **Có được phép sử dụng giá trị null không?**
	Are there any file types that are allowed to be uploaded? **Có loại tập tin nào được phép tải lên không?**
Thông báo	Are general error notifications consistent across the entire project? **Các thông báo lỗi chung có nhất quán trong toàn bộ dự án không?**

	Is the error notification reported correctly? **Thông báo lỗi có được báo cáo chính xác không?**
	Error notifications are written in natural language, easy to understand for users? **Thông báo lỗi được viết bằng ngôn ngữ tự nhiên, dễ hiểu cho người dùng?**
	Does the error notification do not blame the user and suggest how to fix it? **Thông báo lỗi có đổ lỗi cho người dùng và gợi ý cách khắc phục không?**
	When an action affecting the system (edit, delete) is performed, is the user asked again for confirmation? **Khi thực hiện một hành động ảnh hưởng đến hệ thống (chỉnh sửa, xóa), người dùng có được yêu cầu xác nhận lại không?**
	Notifications for exceptions to conditions **Thông báo về các trường hợp ngoại lệ đối với các điều kiện**
	Notification for invalid input **Thông báo nhập liệu không hợp lệ**
	Notice for boundary conditions **Thông báo về điều kiện biên giới**
Trạng thái	Are the statuses displayed correctly? **Các trạng thái có được hiển thị đúng không?**

	Are the status colors different so users can easily distinguish them? **Màu sắc trạng thái có khác nhau để người dùng có thể dễ dàng phân biệt không?**
Database	
Add	When a new record is successfully added, it is displayed below the list records? **Khi một bản ghi mới được thêm thành công, nó sẽ được hiển thị bên dưới danh sách bản ghi?**
	When adding a new record that is missing required fields, is there a warning to the user? **Khi thêm một bản ghi mới thiếu các trường bắt buộc, có cảnh báo nào cho người dùng không?**
	When the new addition is failed, the record will not be saved to the DB? **Khi việc bổ sung mới không thành công, bản ghi sẽ không được lưu vào DB?**
	When the new addition is successful, the record will be saved to the DB? **Khi việc thêm mới thành công, bản ghi sẽ được lưu vào DB?**
Delete	Before performing an operation that affects the system (for example, deleting a record), is the user asked for confirmation? **Trước khi thực hiện thao tác ảnh hưởng đến hệ thống (ví dụ: xóa bản ghi), người dùng có được yêu cầu xác nhận không?**
	When deleting a record that is related to a record of another table, will the data in the relevant record be updated? **Khi xóa một bản ghi có liên quan đến một bản ghi trong bảng khác, dữ liệu trong bản ghi có liên quan có được cập nhật không?**
	On successful deletion, the record is not deleted from the DB but only changed status? **Khi xóa thành công, bản ghi không bị xóa khỏi DB mà chỉ thay đổi trạng thái?**
Edit	When the record being edited is duplicated with another record already in the system, does the system warn users? **Khi bản ghi đang được chỉnh sửa bị trùng lặp với một bản ghi khác đã có trong hệ thống, hệ thống có cảnh báo người dùng không?**
	When the record is successfully edited, a notification is displayed to the user? **Khi bản ghi được chỉnh sửa thành công, thông báo sẽ hiển thị cho người dùng?**

	When editing a record that is related to a record in another table, is the data in the relevant record updated? **Khi chỉnh sửa một bản ghi có liên quan đến một bản ghi trong bảng khác, dữ liệu trong bản ghi có liên quan có được cập nhật không?**

	When the record is successfully edited, the data is updated in the DB? **Khi bản ghi được chỉnh sửa thành công, dữ liệu sẽ được cập nhật trong DB?**
General	Is there a mechanism to handle simultaneous access to a record for updating? **Có cơ chế nào để xử lý việc truy cập đồng thời vào bản ghi để cập nhật không?**
	Rollback data when a transaction fails? **Hoàn nguyên dữ liệu khi giao dịch không thành công?**
	Implemented trim space at the beginning and end of data fields when saving to DB. When entering a text value, is there a space value at the beginning and end? **Đã triển khai khoảng cách cắt ở đầu và cuối các trường dữ liệu khi lưu vào DB. Khi nhập một giá trị văn bản, có giá trị khoảng cách ở đầu và cuối không?**
	Are combo box values saved as ids: 0, 1, 2... **Các giá trị của hộp kết hợp có được lưu dưới dạng id: 0, 1, 2... không?**
Major	
	Covered all valid scenarios according to the specification? **Đã bao gồm tất cả các tình huống hợp lệ theo thông số kỹ thuật chưa?**
	Are all invalids' scenarios covered according to the specification? **Có phải tất cả các tình huống không hợp lệ đều được đề cập theo thông số kỹ thuật không?**
	Has the implementation process optimized the number of operations? **Quá trình triển khai có tối ưu hóa được số lượng thao tác không?**
	Is the process easy to understand and use? **Quá trình này có dễ hiểu và dễ sử dụng không?**
	Is the function/interface easy to edit in case the user needs to change/add new? **Chức năng/giao diện có dễ chỉnh sửa khi người dùng cần thay đổi/thêm mới không?**