**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

****

**BÀI KIỂM TRA THƯỜNG XUYÊN 1**

**HỌC PHẦN: KIỂM THỬ PHẦN MỀM**

Giáo viên hướng dẫn: Thầy Hoàng Quang Huy

Người thực hiện: Nguyễn Văn Huấn

Mã sinh viên: 2020600491

Lớp: **20222IT6013003**

**Câu 1: Trình bày về kiểm thử phần mềm?**

* 1. **Kiểm thử phần mềm là gì?**

Là quá trình đánh giá chất lượng, tính năng, hiệu suất và độ tin cậy của một phần mềm trước khi nó được triển khai ra thị trường hoặc sử dụng bởi người dùng cuối.

* *Mục đích của kiểm thử phần mềm là xác định các lỗi, lỗ hổng hoặc các yêu cầu còn thiếu so với yêu cầu thực tế.*
  1. **Các loại kiểm thử?**

Được phân làm ba loại:

* Types of Software Testing *(Kiểm thử chức năng)*
* Functional Testing *(Kiểm thử phi chức năng)*
* Maintenance Testing *(Kiểm thử sự bảo trì)*

*Hình 1.1 Các loại kiểm thử*

* 1. **Các nguyên lý kiểm thử**
* Mọi kiểm tra đều có thể truy vết đến yêu cầu khách hàng.
* Mọi kiểm tra nên được lập kế hoạch trước khi thực hiện.
* Nguyên lý Perato ứng dụng vào kiểm thử phần mềm.
* Việc kiểm thử nên được thực hiện trong phạm vi nhỏ rồi hướng tới phạm vi lớn.
* Kiểm thử toàn bộ là không thể.
* Việc kiểm thử nên được thực hiện bởi 1 bên thứ 3 độc lập.

**1.4 Vai trò của kiểm thử phần mềm là gì?**

* **Cost-Effective (Hiệu quả về chi phí):** Đây là một trong những lợi thế quan trọng của kiểm thử phần mềm. Thử nghiệm bất kỳ dự án CNTT nào đúng hạn sẽ giúp bạn tiết kiệm tiền trong thời gian dài. Trong trường hợp nếu các lỗi bắt gặp trong giai đoạn kiểm thử phần mềm trước đó, thì sẽ tốn ít chi phí hơn để sửa.
* **Security (Bảo mật):** Đây là lợi ích dễ bị tổn thương và nhạy cảm nhất của kiểm thử phần mềm. Mọi người đang tìm kiếm các sản phẩm đáng tin cậy. Nó giúp loại bỏ rủi ro và các vấn đề sớm hơn.
* **Product quality (Chất lượng sản phẩm):** Là yêu cầu tất yếu của bất kỳ sản phẩm phần mềm nào. Kiểm tra đảm bảo một sản phẩm chất lượng được cung cấp cho khách hàng.
* **Customer Satisfaction (Sự hài lòng của khách hàng):** Mục đích chính của bất kỳ sản phẩm nào là mang lại sự hài lòng cho khách hàng của họ. Kiểm tra UI/UX đảm bảo trải nghiệm người dùng tốt nhất.

**Tại sao kiểm thử phần mềm lại quan trọng và cần thiết?**

* **Tăng độ tin cậy của phần mềm:** Kiểm thử phần mềm giúp xác định lỗi và sửa chúng trước khi phần mềm được triển khai, do đó tăng độ tin cậy của phần mềm.
* **Giảm thiểu rủi ro:** Kiểm thử phần mềm giúp xác định các lỗi và sửa chúng, do đó giảm thiểu rủi ro khi triển khai phần mềm.
* **Tiết kiệm thời gian và chi phí:** Kiểm thử phần mềm giúp phát hiện và sửa chữa các lỗi trước khi phần mềm triển khai, giúp tiết kiệm thời gian và chi phí so với việc phát hiện và sửa lỗi sau khi triển khai phần mềm.
* **Đảm bảo tính tương thích:** Kiểm thử phần mềm giúp đảm bảo tính tương thích của phần mềm với các hệ thống và ứng dụng khác.
* **Đảm bảo sự thoả mãn của khách hàng:** Kiểm thử phần mềm giúp đảm bảo rằng phần mềm thỏa mãn các yêu cầu của khách hàng và đáp ứng các mong đợi của họ.

**1.5 Quy trình của kiểm thử**

Thường được chia làm 6 bước cơ bản như sau:

1. Lập kế hoạch kiểm thử phần mềm (Test Planning): Kế hoạch kiểm thử phần mềm sẽ xác định phương pháp, quy trình và tài nguyên cho hoạt động kiểm thử phần mềm. Bước này cũng ghi chép các yêu cầu, tài liệu và mục tiêu của quá trình kiểm thử.
2. Xác định yêu cầu kiểm thử (Test Requirements): Bước này nhằm xác định và phân tích các yêu cầu kiểm thử, điều kiện kiểm thử và tiêu chí đánh giá cho phần mềm.
3. Thiết kế kiểm thử (Test Design): Thiết kế kiểm thử sử dụng yêu cầu kiểm thử và phân tích các kịch bản kiểm thử để thiết kế các ca kiểm thử cho phần mềm.
4. Thực hiện các ca kiểm thử (Test Execution): Bước này yêu cầu các lập trình viên hoặc kiểm thử viên tiến hành thực hiện các ca kiểm thử đã thiết kế để đánh giá hiệu năng của phần mềm.
5. Phân tích kết quả (Test Result Analysis): Bước này nhằm phân tích kết quả kiểm thử để đánh giá phản hồi cho kiểm thử và có các phản niêm yếu tố trong phần mềm.
6. Báo cáo (Test Reporting): Bước cuối cùng là tạo ra các báo cáo về quá trình kiểm thử phần mềm cho các bên liên quan, bao gồm các kết quả kiểm thử và phản hồi cho quá trình kiểm thử.

**1.6 Chiến lược kiểm thử**

* **Kiểm thử đơn vị (Unit testing):** kiểm thử chức năng của từng đơn vị code (hàm, phương thức, lớp,...).
* **Kiểm thử tích hợp (Integration testing):** kiểm tra kết quả hoạt động khi nhiều đơn vị code kết hợp với nhau.
* **Kiểm thử hệ thống (System testing):** kiểm tra toàn bộ hệ thống để đảm bảo tính đầy đủ và đúng đắn của các chức năng và tính năng, bao gồm cả các kiểm thử tương tác với người dùng.
* **Kiểm thử chấp nhận (Acceptance testing):** kiểm tra để đảm bảo rằng phần mềm đã thỏa mãn các yêu cầu và kỳ vọng của người dùng.
  1. **Các kỹ thuật về kiểm thử**
* Kỹ thuật kiểm thử hộp đen
* Kỹ thuật kiểm thử hộp trắng

**Câu 2: Mô hình V trong kiểm thử phần mềm**

**2.1 Mô hình V là gì?**

**Mô hình V** là một mô hình SDLC có tính kỷ luật cao, có giai đoạn thử nghiệm song song với từng giai đoạn phát triển. Mô hình V là một phần mở rộng của mô hình thác nước trong đó quá trình phát triển và thử nghiệm phần mềm được thực hiện theo cách tuần tự. Nó được gọi là mô hình xác thực hoặc xác minh.

Đánh giá yêu cầu

Kiểm tra hệ thống

Kiểm tra tích hợp

Thiết kế cấp cao

Kiểm tra đơn vị

Thiết kế cấp thấp

Coding

*Hình 2.1 Mô hình V trong kiểm thử phần mềm*

Thuật ngữ phần mềm chính

* **SDLC:** là Vòng đời phát triển phần mềm. Đó là chuỗi các hoạt động được Nhà phát triển thực hiện để thiết kế và phát triển phần mềm chất lượng cao.
* **STLC:**  là Vòng đời kiểm thử phần mềm. Nó bao gồm một loạt các hoạt động được thực hiện bởi Người kiểm tra theo phương pháp để kiểm tra sản phẩm phần mềm của bạn.
* **Mô hình thác nước:** là một mô hình tuần tự được chia thành các giai đoạn khác nhau của hoạt động phát triển phần mềm. Mỗi giai đoạn được thiết kế để thực hiện các hoạt động cụ thể. Giai đoạn thử nghiệm trong mô hình thác nước chỉ bắt đầu sau khi triển khai hệ thống xong.

Bên trái của mô hình là Vòng đời phát triển phần mềm – **SDLC**

**Phân tích yêu cầu**:

Trong giai đoạn này các yêu cầu được thu thập, phân tích và nghiên cứu. Ở giai đoạn này việc hệ thống chạy như thế nào không quan trọng, quan trọng là hệ thống có những chức năng gì.

* Hoạt động xác minh:Đánh giá yêu cầu (Requirements review).
* Hoạt động xác nhận: Tạo test case UAT (User acceptance test- kiểm thử chấp nhận) = Đầu ra cần có: Tài liệu hiểu về yêu cầu, UAT test case.

**Thiết kế cấp cao**

Trong giai đoạn này, high level design của phần mềm được xây dựng. Nhóm sẽ nghiên cứu và điều tra về các yêu cầu có thể được thực hiện như thế nào. Tính khả thi về mặt kỹ thuật của yêu cầu cũng được tìm hiểu. Nhóm cũng tìm hiểu về các mô-đun sẽ được tạo / phụ thuộc, nhu cầu phần cứng / phần mềm

* Hoạt động xác minh: Đánh giá thiết kế
* Hoạt động xác nhận: Tạo test plan và test case, tạo ma trân truy vết
* Đầu ra cần có: System test cases, Feasibility reports, System test plan, tài liệu về yêu cầu phần cứng và các mô đun, ...

**Thiết kế kiến trúc**

Trong giai đoạn này, dựa trên thiết kế mức cao, kiến trúc phần mềm được tạo ra. Các mô-đun, mối quan hệ và sự phụ thuộc của họ, sơ đồ kiến trúc, bảng cơ sở dữ liệu, chi tiết về công nghệ đều được hoàn tất trong giai đoạn này.

* Hoạt động xác minh: Đánh giá thiết kế
* Hoạt động xác nhận: Kế hoạch thử nghiệm tích hợp và các trường hợp thử nghiệm.
* Đầu ra cần có: Tài liệu thiết kế, Kế hoạch kiểm thử tích hợp và các trường hợp thử nghiệm, Thiết kế bảng cơ sở dữ liệu,...

**Thiết kế cấp thấp**

Trong giai đoạn này mỗi mô-đun hoặc các thành phần phần mềm đều được thiết kế riêng. Các method, class, giao diện, các kiểu dữ liệu vv đều được hoàn tất trong giai đoạn này.

* Hoạt động xác minh: Đánh giá thiết kế
* Hoạt động xác nhận: Tạo và xem xét các trường hợp kiểm tra đơn vị.
* Đầu ra cần có: Các đơn vị kiểm tra đơn vị

**Code**

Trong giai đoạn này, code được thực hiện.

* Hoạt động xác minh: Xem xét mã, kiểm tra các trường hợp kiểm tra
* Hoạt động xác nhận: Tạo các trường hợp kiểm tra chức năng.
* Đầu ra cần có: các trường hợp thử nghiệm, danh sách kiểm tra xem lại.

Bên phải của mô hình là Vòng đời kiểm thử phần mềm – **STLC**

**Kiểm tra đơn vị**

Trong giai đoạn này tất cả unit test case đã được tạo ra trong giai đoạn thiết kế cấp thấp sẽ được thực hiện.

* Unit test là một kỹ thuật kiểm tra hộp trắng, nơi một đoạn code được viết, nó sẽ tích hợp một phương pháp (hoặc một đoạn code khác) để kiểm tra xem đoạn code có cho kết quả mong muốn hay không. Thử nghiệm này về cơ bản được thực hiện bởi nhóm phát triển. Trong trường hợp có bất thường, bug được ghi lại và theo dõi.
* Đầu ra cần có: Kết quả thực hiện unit test

**Kiểm tra tích hợp**

Trong giai đoạn này các trường hợp kiểm thử tích hợp được thực hiện đã được tạo ra trong giai đoạn thiết kế kiến trúc. Trong trường hợp có bất kỳ dị thường, bug được ghi lại và theo dõi.

* Kiểm thử tích hợp (Integration Testing): Kiểm thử tích hợp là một kỹ thuật mà đơn vị kiểm tra là mô-đun được tích hợp và kiểm tra xem các mô-đun tích hợp vào có cho kết quả mong đợi không. Nói một cách đơn giản, nó xác nhận xem các thành phần của ứng dụng có hoạt động với nhau như mong đợi hay không.
* Đầu ra cần có: Các kết quả kiểm tra tích hợp.

**Kiểm tra hệ thống**

Trong giai đoạn này, kiểm thử chức năng và phi chức năng được thực hiện. Nói cách khác, việc kiểm tra thực tế hoạt động của ứng dụng diễn ra ở đây. Lỗi được phát hiện và theo dõi để sửa.

Đầu ra cần có: Kết quả kiểm tra, Các bản ghi kiểm tra, báo cáo lỗi, báo cáo tóm tắt kiểm tra và các ma trận truy xuất cập nhật.

**Thử nghiệm chấp nhận của người dùng**:

Thử nghiệm chấp nhận về cơ bản liên quan đến việc kiểm tra các yêu cầu business. Ở đây kiểm tra được thực hiện để xác nhận rằng các yêu cầu kinh doanh được đáp ứng trong môi trường người dùng.

Đầu ra cần có: Kết quả UAT, Ma trận độ bao phủ business được cập nhật.

**2.2 Khi nào sử dụng mô hình V**

**Mô hình V** được áp dụng khi:

* Yêu cầu được xác định rõ ràng và không mơ hồ
* Tiêu chí chấp nhận được xác định rõ ràng.
* Dự án có quy mô vừa và nhỏ.
* Công nghệ và công cụ được sử dụng không thường xuyên thay đổi.

2.3 Ví dụ để hiểu mô hình V

Chẳng hạn, bạn được giao nhiệm vụ phát triển phần mềm cho khách hàng. Chúng ta sẽ không nhắc đến nền tảng kỹ thuật mà sẽ đưa ra giả thiết về các bước thực hiện để hoàn thành nhiệm vụ.

|  |  |
| --- | --- |
| **Các giai đoạn trong chu kỳ phát triển phần mềm** | **Các hoạt động được thực hiện trong từng giai đoạn** |
| Thu thập yêu cầu | Thu thập thông tin chi tiết, thông số kỹ thuật của phần mềm, mong muốn của khách hàng |
| Thiết kế hệ thống | Lên kế hoạch lựa chọn cơ sở dữ liệu (MySQL, Oracle,…) và ngôn ngữ lập trình (PHP, Java,…) |
| Giai đoạn xây dựng | Giai đoạn này sẽ triển khai thực hiện mã hóa code |
| Giai đoạn kiểm thử | Xác minh xem nó được xây dựng đúng theo thông số kỹ thuật được cung cấp bởi khách hàng hay không |
| Giai đoạn triển khai | Triển khai ứng dụng, phần mềm trong môi trường thật |
| Giai đoạn bảo trì | Nâng cấp, bảo trì hoặc thay đổi code theo yêu cầu của khách hàng |

**2.4 Ưu điểm và nhựơc điểm khi sử dụng mô hình V:**

**1. Ưu điểm:**

* Quá trình phát triển và quy trình quản lý có tính tổ chức và hệ thống
* Hoạt động tốt cho các dự án có quy mô vừa và nhỏ.
* Kiểm tra bắt đầu từ khi bắt đầu phát triển vì vậy sự mơ hồ được xác định ngay từ đầu.
* Dễ dàng quản lý vì mỗi giai đoạn có các mục tiêu và mục tiêu được xác định rõ ràng.

**2. Nhược điểm:**

* Không thích hợp cho các dự án lớn và phức tạp
* Không phù hợp nếu các yêu cầu thường xuyên thay đổi.
* Không có phần mềm làm việc được sản xuất ở giai đoạn trung gian.
* Không có điều khoản cho việc phân tích rủi ro nên có sự không chắc chắn và có tính rủi ro.

**Câu 3: Kiểm thử Agile**

**3.1 Kiểm thử Agile là gì?**

Kiểm thử Agile là thực hiện kiểm thử tuân theo các quy tắc và nguyên tắc của phát triển phần mềm Agile. Không giống như phương pháp Waterfall, Agile Testing có thể bắt đầu khi bắt đầu dự án với sự tích hợp liên tục giữa phát triển và kiểm thử. Phương pháp Agile Testing không tuần tự (nghĩa là nó chỉ được thực thi sau giai đoạn phát triển) mà là liên tục.

**3.2 Nguyên tắc kiểm thử Agile**

Dưới đây là các Nguyên tắc thiết yếu của **Kiểm thử Agile**:

* Trong mô hình thử nghiệm Agile này, phần mềm hoạt động được là thước đo tiến độ chính.
* Các nhóm tự tổ chức có thể đạt được kết quả tốt nhất.
* Cung cấp phần mềm có giá trị sớm và liên tục là ưu tiên cao nhất của chúng tôi.
* Các nhà phát triển phần mềm phải hành động để thu thập hàng ngày trong suốt dự án.
* Agile Testing đảm bảo rằng sản phẩm cuối cùng đáp ứng mong đợi của doanh nghiệp bằng cách cung cấp phản hồi liên tục.
* Trong quy trình Agile Test, chúng ta cần thực hiện kiểm thử trong khi triển khai.
* Quá trình thử nghiệm trong Agile nên hoạt động theo tốc độ phát triển nhất quán
* Cung cấp những phản ánh thường xuyên về cách trở nên hiệu quả hơn.
* Các kiến ​​trúc, yêu cầu và thiết kế tốt nhất xuất hiện từ các nhóm tự tổ chức.
* Mỗi khi nhóm họp, nhóm sẽ xem xét và điều chỉnh hành vi của mình để trở nên hiệu quả hơn.
* Trò chuyện trực tiếp với nhóm phát triển là phương pháp hiệu quả và hiệu quả nhất để truyền đạt thông tin trong nhóm.

**3.3 Vòng đời kiểm thử Agile**

Vòng đời Thử nghiệm linh hoạt được hoàn thành theo năm giai đoạn khác nhau, như chúng ta có thể thấy trong hình ảnh sau:

*Hình 3.1 Vòng đời kiểm thử Agile*

Dưới đây là các bước kiểm tra quy trình Agile:

**1. Đánh giá tác động**: Trong giai đoạn đầu này, chúng tôi thu thập thông tin đầu vào từ các bên liên quan và người dùng. Giai đoạn này còn được gọi là giai đoạn phản hồi, vì nó hỗ trợ kiểm tra thiết lập các mục tiêu cho vòng đời tiếp theo.

**2. Lập kế hoạch thử nghiệm Agile**: Đây là giai đoạn thứ hai của vòng đời thử nghiệm Agile, trong đó tất cả các bên liên quan cùng nhau lên kế hoạch cho lịch trình của quy trình thử nghiệm và các sản phẩm bàn giao.

**3. Sẵn sàng phát hành**: Ở giai đoạn này, chúng tôi xem xét các tính năng đã được phát triển/Triển khai đã sẵn sàng hoạt động hay chưa. Trong giai đoạn này, người ta cũng quyết định cái nào cần quay lại giai đoạn phát triển trước đó.

**4. Scrums hàng ngày**: Giai đoạn này bao gồm mọi cuộc họp đứng vào buổi sáng để nắm bắt tình hình kiểm thử và đặt mục tiêu cho cả ngày.

**5. Kiểm tra Đánh giá Agility**: Giai đoạn cuối cùng của vòng đời Agile là Cuộc họp Đánh giá Agility. Nó liên quan đến các cuộc họp hàng tuần với các bên liên quan để thường xuyên đánh giá và đánh giá tiến độ so với các mục tiêu.

**3.4 Kế hoạch kiểm thử Agile**

Các kế hoạch thử nghiệm điển hình trong **Agile** bao gồm:

* Phạm vi thử nghiệm
* Các chức năng mới đang được thử nghiệm
* Mức độ hoặc Các loại thử nghiệm dựa trên độ phức tạp của tính năng
* Kiểm tra tải và hiệu suất
* Xem xét cơ sở hạ tầng
* Kế hoạch giảm thiểu hoặc rủi ro
* Nguồn lực
* Chuyển giao và các cột mốc

**3.5 Chiến lược kiểm thử Agile**

Vòng đời kiểm thử Agile trải qua bốn giai đoạn

**Giai đoạn 1: Iteration 0**

a) Thiết lập nghiệp vụ cho dự án

b) Thiết lập các điều kiện ranh giới và phạm vi dự án

c) Vạch ra các yêu cầu chính và các trường hợp sử dụng sẽ thúc đẩy sự đánh đổi thiết kế

d) Phác thảo một hoặc nhiều kiến trúc ứng viên

e) Xác định rủi ro

f) Dự toán chi phí và chuẩn bị một dự án sơ bộ

**Giai đoạn 2: Construction Iterations**

Giai đoạn thứ hai của phương pháp kiểm thử nhanh là Lặp lại xây dựng, phần lớn kiểm thử xảy ra trong giai đoạn này. Giai đoạn này được quan sát như một tập hợp các lần lặp lại để xây dựng phần gia tăng của giải pháp. Để làm được điều đó, trong mỗi lần lặp lại, nhóm thực hiện kết hợp các phương pháp thực hành từ XP, Scrum, mô hình Agile và dữ liệu nhanh, v.v.

**Giai đoạn 3: Release End Game Or Transition Phase**

Mục tiêu của “Release, End Game” là triển khai thành công hệ thống của bạn vào sản xuất. Các hoạt động bao gồm trong giai đoạn này là đào tạo người dùng cuối, người hỗ trợ và người vận hành. Ngoài ra, nó bao gồm tiếp thị việc phát hành sản phẩm, sao lưu và phục hồi, hoàn thiện hệ thống và tài liệu người dùng.

Giai đoạn thử nghiệm phương pháp nhanh cuối cùng bao gồm thử nghiệm toàn bộ hệ thống và thử nghiệm chấp nhận. Để hoàn thành giai đoạn thử nghiệm cuối cùng của bạn mà không gặp bất kỳ trở ngại nào, bạn phải kiểm tra sản phẩm nghiêm ngặt hơn khi nó đang trong quá trình lặp lại xây dựng. Trong khi trò chơi kết thúc, người kiểm tra sẽ làm việc với các câu chuyện lỗi của nó.

**Giai đoạn 4: Production**

Sau giai đoạn xuất xưởng, sản phẩm sẽ chuyển sang giai đoạn sản xuất.

**Các góc phần tư thử nghiệm Agile**

Các góc phần tư kiểm thử agile phân tách toàn bộ quá trình thành bốn Góc phần tư và giúp hiểu cách kiểm tra nhanh được thực hiện.

**Agile Quadrant I**

Chất lượng mã nội bộ là trọng tâm chính trong góc phần tư này và nó bao gồm các trường hợp thử nghiệm theo hướng công nghệ và được triển khai để hỗ trợ nhóm, nó bao gồm

1. Bài kiểm tra đơn vị
2. thử nghiệm thành phần

**Agile Quadrant II**

Nó chứa các trường hợp thử nghiệm theo hướng kinh doanh và được triển khai để hỗ trợ nhóm. Góc phần tư này tập trung vào các yêu cầu. Loại thử nghiệm được thực hiện trong giai đoạn này là

1. Kiểm tra các ví dụ về các tình huống và quy trình công việc có thể xảy ra
2. Kiểm tra trải nghiệm người dùng chẳng hạn như nguyên mẫu
3. Thử nghiệm theo cặp

**Agile Quadrant III**

Góc phần tư này cung cấp phản hồi cho góc phần tư một và hai. Các trường hợp kiểm thử có thể được sử dụng làm cơ sở để thực hiện kiểm thử tự động hóa. Trong góc phần tư này, nhiều vòng đánh giá lặp đi lặp lại được thực hiện để tạo niềm tin vào sản phẩm. Loại thử nghiệm được thực hiện trong góc phần tư này là

1. Kiểm tra khả năng sử dụng
2. Thử nghiệm thăm dò
3. Ghép nối thử nghiệm với khách hàng
4. Thử nghiệm cộng tác
5. Kiểm tra sự chấp nhận của người dùng

**Agile Quadrant IV**

Góc phần tư này tập trung vào các yêu cầu phi chức năng như hiệu suất, bảo mật, độ ổn định. Với sự trợ giúp của góc phần tư này, ứng dụng được tạo ra để cung cấp các chất lượng phi chức năng và giá trị mong đợi.

1. Kiểm tra phi chức năng như kiểm tra căng thẳng và hiệu suất
2. Kiểm tra bảo mật liên quan đến xác thực và hack
3. Kiểm tra cơ sở hạ tầng
4. Thử nghiệm di chuyển dữ liệu
5. Kiểm tra khả năng mở rộng
6. Kiểm tra tải

**Câu 4: Trình bày kiểm thử bảo mật web, cho ví dụ?**

**4.1 Security testing là gì?**

Security Testing là việc tìm kiếm tất cả cá lỗ hổng có thể và điểm yếu của hệ thống mà có thể dẫn đến mất thông tin trong tay nhân viên hoặc người ngoài của tổ chức. Security Testing rất quan trọng trong ngành công nghiệp CNTT để bảo vệ dữ liệu của tất cả các phương tiện.

**4.2 Mục đích của Security Testing?**

Mục đích của Security Testing là để xác định các mối đe dọa trong hệ thống và đo lường rủi ro tiềm năng của nó. Nó cũng giúp trong việc phát hiện các nguy cơ bảo mật có thể có trong hệ thống và giúp các developer trong việc sửa chữa những vấn đề này thông qua code

Phương pháp

Dưới đây là các phương pháp khác nhau cho Security Testing:

* Tiger Box: được thực hiện trên một máy tính xách tay trong đó có một bộ sưu tập của các hệ điều hành và các công cụ hack. Thử nghiệm này giúp kiểm tra sự thâm nhập và kiểm tra bảo mật để tiến hành đánh giá các lỗ hổng và các cuộc tấn công.
* Black Box: Tester được ủy quyền để làm thử nghiệm trên tất cả mọi thứ về các cấu trúc liên kết mạng lưới và công nghệ.
* Grey Box: là sự kết hợp của mô hình black box và white box

**4.3 Các loại Security testing**

Có bảy loại chính trong Security testing :

| Types of Security Testing | Description |
| --- | --- |
| Vulnerability Scanning | Được thực hiện thông qua phần mềm tự động quét hệ thống chống lại lỗ hổng đã biết |
| Security Scanning | Nó liên quan đến việc xác định mạng và hệ thống những điểm yếu, sau đó cung cấp các giải pháp để giảm thiểu những rủi ro này.Có thể được thực hiện cho cả hai chức năng quét bằng tay và tự động |
| Penetration Testing | Đây là loại thử nghiệm mô phỏng một cuộc tấn công từ hacker. Kiểm tra này bao gồm việc phân tích một hệ thống đặc biệt để kiểm tra các lỗ hổng tiềm năng với một nỗ lực hacking bên ngoài. |
| Risk Assessment | Kiểm tra này bao gồm việc phân tích các rủi ro an ninh quan sát trong tổ chức.Thử nghiệm này đề nghị kiểm soát và các biện pháp để giảm thiểu rủi ro |
| Security Auditing | Đây là kiểm tra nội bộ của các ứng dụng và hệ thống điều hành cho lỗ hổng bảo mật.Kiểm toán cũng có thể được thực hiện thông qua đường bằng cách kiểm tra dòng mã |
| Posture Assessment | Kết hợp Security Scanning và Ethical Hacking và Risk Assessment cho thấy một an ninh tổng thể của một tổ chức. |
| Ethical Hacking | Nó hack một hệ thống phần mềm tổ chức, mục đích là để lộ lỗ hổng bảo mật trong hệ thống |

**4.4 Tích hợp các quy trình bảo mật với các SDLC**

| Các giai đoạn của SDLC | Quy trình bảo mật |
| --- | --- |
| Yêu cầu | Phân tích bảo mật cho các yêu cầu và kiểm tra tình trạng lạm dụng / trường hợp sử dụng sai |
| Thiết kế | Phân tích rủi ro bảo mật cho thiết kế. Xây dựng kế hoạch kiểm tra bao gồm các bài kiểm tra an ninh |
| Coding & Unit testing | Kiểm thử hộp trắng |
| Intergration Testing | Black Box Testing |
| System Testing | Black Box Testing và Vulnerability Scanning |
| Thực hiện | Penetration Testing, Vulnerability Scanning |
|  |  |
| Support | Phân tích tác động |

Các Test scenarios mẫu cho Security Testing:

Các kịch bản thử nghiệm mẫu để cung cấp một cái nhìn thoáng hơn trong Security Testing

* Mật khẩu phải ở trong định dạng mã hóa
* Ứng dụng hoặc hệ thống không nên cho phép người dùng không hợp lệ
* Kiểm tra thời gian cookie và session cho ứng dụng
* Đối với các trang web tài chính, nút Back của Browser lại không nên được hoạt động.

**4.5 Ví dụ về kiểm thử bảo mật**

**4.5.1 Công cụ sử dụng OWASP ZAP**

**OWASP ZAP –  Zed Attack Proxy**  là một công cụ kiểm tra lỗ hổng bảo mật giúp các lập trình viên và chuyên gia bảo mật phát hiện ra các lỗ hổng của Website. Bạn có thể thực hiện nhiều tác vụ khác nhau thông qua OWASP ZAP, bao gồm quét các request web, sử dụng danh sách dictionary để tìm các filevà thư mục trên máy chủ web và các trang web thu thập dữ liệu để tìm tất cả các liên kết và URL. Ngoài ra, bạn có thể chặn, xem, sửa đổi và chuyển tiếp các request web giữa trình duyệt và ứng dụng web bằng OWASP ZAP.

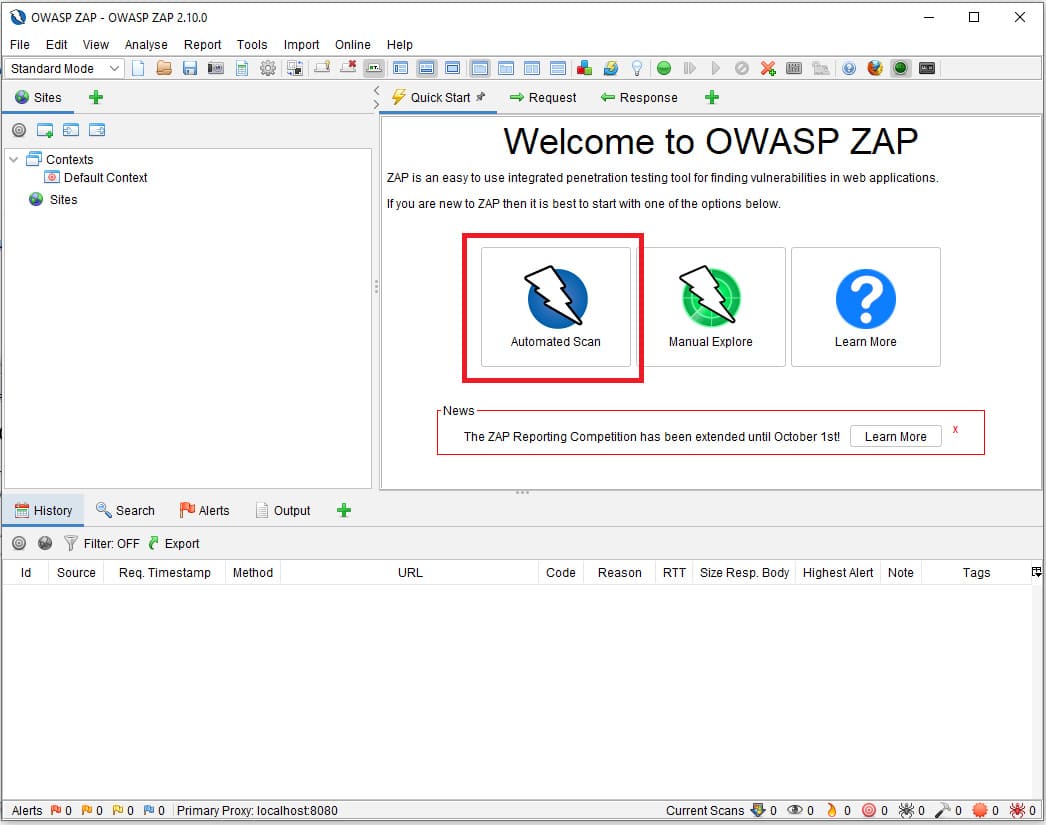
Quá trình **quét OWASP ZAP** có thể tiết lộ các vấn đề liên quan đến SQL injection, broken authentication, lộ dữ liệu nhạy cảm, kiểm soát truy cập bị hỏng, cấu hình sai bảo mật, Cross-Site Scripting(XSS), giải mã không an toàn, các thành phần có lỗ hổng đã biết và bất kỳ headers bảo mật nào bị thiếu.

**4.5.2 Hướng dẫn sử dụng Owasp Zap**

Sau khi tải xong, bạn sẽ nhận được một màn hình trông giống như ảnh dưới.

Trong chế độ xem ‘Quick Start’ chính, bạn có thể chọn giữa ‘Automated Scan’ và ‘Manual Explore’’.

Đối với hướng dẫn này, hãy chọn ‘Automated Scan’.



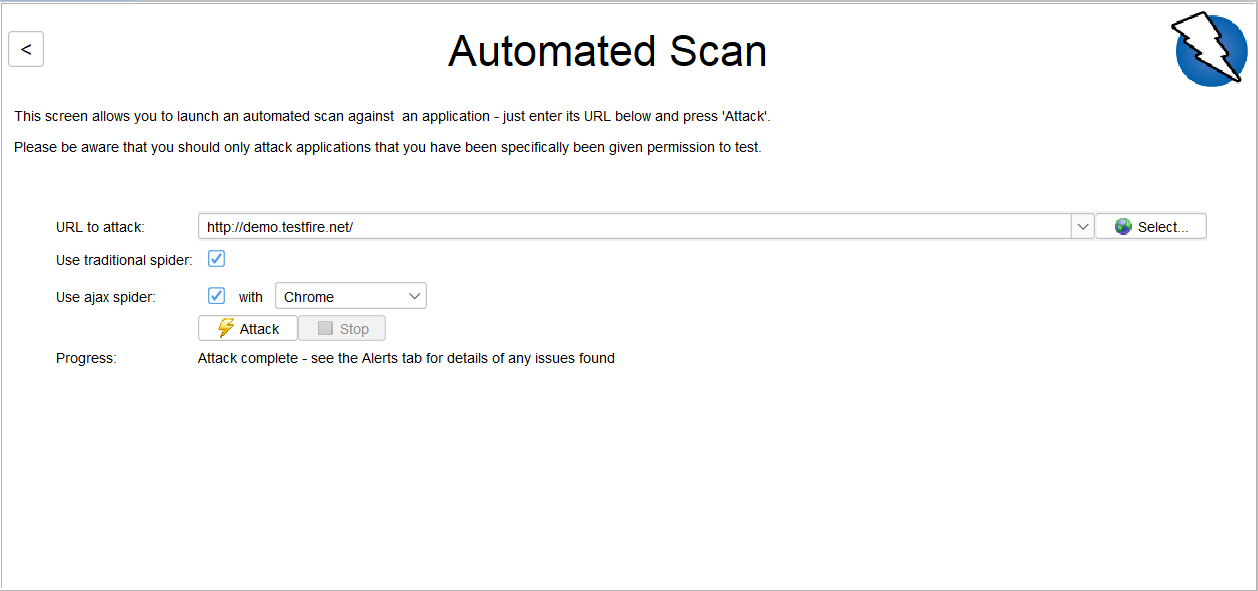
*Hình 4.1 Giao diện sau khi khởi động*

Tùy chọn này sẽ cung cấp cho bạn một trang giống như ảnh dưới. Bạn có thể chạy nó trên ứng dụng của mình bằng cách trỏ URL đích đến máy chủ cục bộ mà ứng dụng của bạn hiện đang chạy.

Đây là một ví dụ về khi ở trong giai đoạn quét

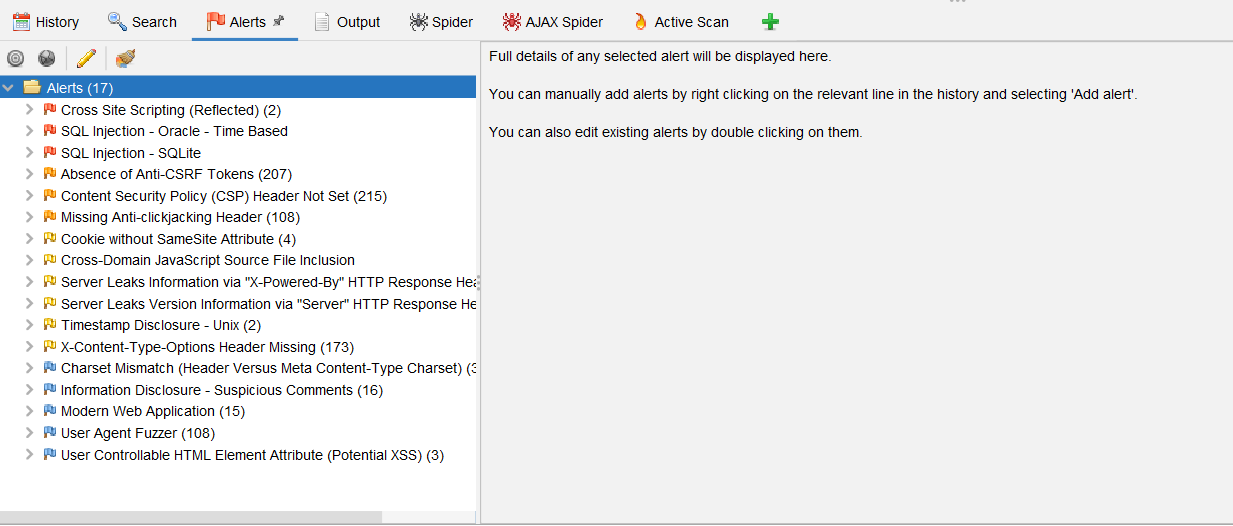
Trang web sử dụng: <http://demo.testfire.net/index.jsp>

Chế độ quét: Quét tự động



*Hình 4.2 Chế độ quét tự động với* ***Owasp Zap***

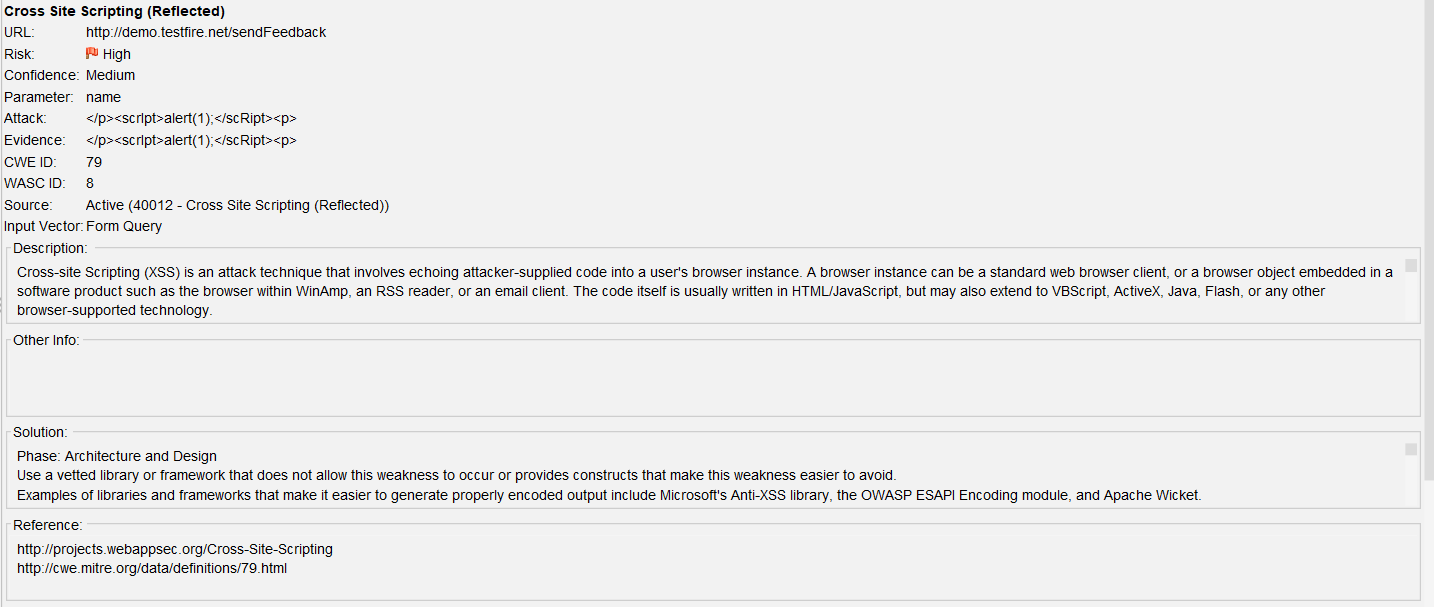
Sau khi hoàn tất, bạn có thể nhấp vào tab ‘Alerts’ để xem tất cả các lỗ hổng tiềm ẩn



*Hình 4.3 Thông báo lỗi sau khi quá trình quét kết thúc*

Dựa trên ví dụ và kết quả quét ở trên, chúng ta có thể thấy rằng các vấn đề được phát hiện là khá nghiêm trọng và phải khắc phục, chẳng hạn như Cross-Site Scripting hay SQL injection

Đây là ảnh chụp màn hình của một trong các cảnh báo được gắn cờ và báo cáo được tạo cho lỗi Cross Site Scripting(Reflected). Ngay ở phía dưới là mô tả chi tiết về lỗi và giải pháp về cách giảm lỗ hổng bảo mật dựa lỗi này.



*Hình 4.4 Thông tin chi tiết về lỗi*