**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ - ĐHQGHN**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**NGUYỄN TIẾN ĐẠT**

**TÌM HIỂU VÀ XÂY DỰNG CÔNG CỤ KIỂM THỬ KHẢ NĂNG TIẾP CẬN CỦA ỨNG DỤNG WEB**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY**

**Ngành: Khoa học máy tính**

**HÀ NỘI - 2022**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ - ĐHQGHN**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**NGUYỄN TIẾN ĐẠT**

**TÌM HIỂU VÀ XÂY DỰNG CÔNG CỤ KIỂM THỬ KHẢ NĂNG TIẾP CẬN CỦA ỨNG DỤNG WEB**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY**

**Ngành: Khoa học máy tính**

**Cán bộ hướng dẫn: TS. Võ Đình Hiếu**

**Cán bộ đồng hướng dẫn: ThS. Nguyễn Bảo Ngọc**

**HÀ NỘI - 2022**

# Lời cảm ơn

Lời đầu tiên, tôi xin kính gửi lời cảm ơn sâu sắc tới thầy TS. Võ Đình Hiếu đã đưa ra đề tài, định hướng nghiên cứu, tận tình hướng dẫn và giúp đỡ tôi trong suốt quá trình thực hiện khóa luận này. Thầy đã tạo điều kiện tổ chức các buổi trao đổi hàng tuần để cung cấp kiến thức mới về kiểm thử khả năng tiếp cận của ứng dụng web cho sinh viên.

Tôi xin cảm ơn thầy Nguyễn Bảo Ngọc đã luôn động viên, giúp đỡ và đưa ra những góp ý hữu ích cho tôi trong quá trình viết và hoàn thành khóa luận này.

Tôi xin chân thành cảm ơn các thầy, cô, cán bộ trong trường Đại học Công nghệ - Đại học Quốc gia Hà Nội, đặc biệt là các thầy cô trong khoa Công nghệ thông tin đã tạo mọi điều kiện thuận lợi nhất cho tôi học tập và nghiên cứu.

Cuối cùng, tôi cũng xin gửi lời cảm ơn sâu sắc đến tất cả các thầy cô, các anh chị, các bạn trong phòng thí nghiệm của bộ môn Công nghệ Phần mềm đã luôn hỗ trợ tôi rất nhiều về kiến thức chuyên môn trong quá trình thực hiện khóa luận này.

# Tóm tắt

**Tóm tắt**: Trong những năm gần đây, ngày càng nhiều ứng dụng web được thiết kế để người khuyết tật có thể truy cập được. Việc kiểm thử khả năng tiếp cận là cần thiết để đánh giá khả năng truy cập của các ứng dụng web này, từ đó giúp các nhà phát triển có thể khắc phục lỗi vi phạm khả năng tiếp cận một cách kịp thời. Tuy nhiên, các công cụ kiểm thử hiện nay vẫn có những hạn chế về cả chất lượng và phạm vi người sử dụng. Các công cụ này chỉ giúp phát hiện các lỗi tiếp cận ứng dụng web mà chưa đưa ra các gợi ý để sửa lỗi. Điều này khiến cho các nhà phát triển gặp khó khăn trong việc khắc phục lại các lỗi tiếp cận của ứng dụng web. Ngoài ra người dùng phải có hiểu biết về khả năng tiếp cận ứng dụng web thì mới có thể xử lý được các vấn đề mà ứng dụng web gặp phải.

Khóa luận này trình bày những phương pháp cải tiến các công cụ kiểm thử khả năng tiếp cận hiện có. Thông qua việc tìm hiểu và phân tích cơ chế hoạt động của các công cụ, khóa luận đưa ra các thay đổi và đề xuất bổ sung thêm chức năng sửa lỗi để cải thiện chất lượng công cụ được phân tích. Công cụ cũng được xây dựng trên nền tảng ứng dụng web hiện đại, các chức năng được đóng gói qua các API, giao tiếp với người dùng thông qua giao diện, giúp người dùng dễ tiếp cận công cụ. Công cụ được đánh giá dựa trên quá trình thử nghiệm mức độ cải thiện của công cụ được triển khai trên một số trang web khác nhau và mức độ hài lòng khi sử dụng công cụ của người dùng. Kết quả thu được cho thấy công cụ này có cải tiến về cả mặt chất lượng và phạm vi người sử dụng công cụ.

**Từ khóa**: kiểm thử khả năng tiếp cận, người khuyết tật, gợi ý sửa lỗi.

**Abstract**

**Abstract**: In recent years, more and more web applications have been designed to be accessible to people with disabilities. It is need to try to test the next Close-up is necessary to assess the accessibility of web applications, thereby helping developers to fix close-up errors quickly. However, current testing tools still have limitations in terms of both quality and range of users. This tool only helps to detect near-faulty web applications without providing suggestions for fixing errors. This makes it difficult for developers to recover from near-faults of web applications. In addition, users must have an understanding of web application accessibility to be able to handle the problems that web applications have to face.

This key is display the method of the process of the improves current accessibility testing tools. Through studying and analyzing the working mechanism of the tools, the thesis makes changes and proposes additional fixes, additional functions to improve the quality of the analyzed tools. The tool is also built on a modern web application platform, the functions are packaged through APIs, communicating with the user through the interface, making it easy for users to access the tool. The tool is evaluated based on testing the improvement of the tool deployed on a number of different websites and users' satisfaction with using the tool. The results obtained for this tool have improved both in terms of quality and in terms of user reach of the tool.

**Keywords**: web accessibility testing, defects, suggested errors.

# Lời cam đoan

Tôi là Nguyễn Tiến Đạt, sinh viên lớp K63-CACLC3, ngành Khoa học máy tính. Tôi xin cam đoan khóa luận này là những nghiên cứu mà tôi tự tìm hiểu. Mọi thông tin tham khảo từ các tài liệu, công trình nghiên cứu liên quan đều được trích trong khóa luận này. Tôi xin chịu hoàn toàn mọi trách nhiệm về những lời cam đoan trên.

Hà Nội, ngày 01 tháng 05 năm 2022

Sinh viên

Đạt

Nguyễn Tiến Đạt

# Danh mục từ viết tắt

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Từ viết tắt | Cụm từ đầy đủ | Cụm từ tiếng việt |
| 1 | WHO | World Health Organization | Tổ chức Y tế Thế giới |
| 2 | SDLC | Software Development Life Cycle | Chu trình phát triển phần mềm |
| 3 | WCAG | Web Content Accessibility Guidelines | Nguyên tắc về tiếp cận nội dung Web |
| 4 | URL | Uniform Resource Locator | Định vị tài nguyên thống nhất |
| 5 | WAI | Web Accessibility Initiative | Sáng kiến khả năng truy cập Web |
| 6 | ISO | International Organization for Standardization | Tổ chức tiêu chuẩn hoá quốc tế |
| 7 | W3C | World Wide Web Consortium | Tổ chức quốc tế cam kết cải thiện web |
| 8 | EIT | Electronic and Information Technology | Công nghệ thông tin và truyền thông |
| 9 | API | Application Programming Interface | Giao diện lập trình ứng dụng |

Danh sách hình vẽ

[Hình 1 . Vòng đời phát triển phần mềm 13](#_Toc11962)

[Hình 2 . Mô hình use-case của công cụ 36](#_Toc14765)

[Hình 3 . Kiến trúc MyChecker 38](#_Toc7281)

[Hình 4 . Flow chart của MyChecker 39](#_Toc7629)

[Hình 5 . Phân tách kết quả phân tích lỗi 41](#_Toc12488)

[Hình 6 . Sửa lỗi vi phạm khả năng tiếp cận 43](#_Toc9920)

[Hình 7 . Giao diện chính 44](#_Toc29512)

[Hình 8 . Giao diện kết quả phân tích lỗi 45](#_Toc6132)

[Hình 9 . Giao diện kết quả sửa lỗi 45](#_Toc16247)

[Hình 10 . Biểu đồ ca sử dụng của công cụ MyChecker 47](#_Toc2752)

[Hình 11 . Mức độ hài lòng thử nghiệm công cụ 48](#_Toc26620)

Danh sách bảng

[Bảng 1 . Nguyên tắc hỗ trợ tiếp cận nội dung web 18](#_Toc22743)

[Bảng 2 . Tiêu chuẩn theo mục 508 19](#_Toc26914)

[Bảng 3 . Kết quả thử nghiệm các công cụ 29](#_Toc12445)

[Bảng 4 . Lỗi liên kết các phần tử web 30](#_Toc8256)

[Bảng 5 . MyChecker giải quyết các trường hợp lỗi trùng lặp 31](#_Toc28759)

[Bảng 6 . MyChecker giải quyết các trường hợp lỗi không rõ ràng 33](#_Toc30646)

[Bảng 7 . Lỗi được sửa bởi MyChecker 34](#_Toc8619)

[Bảng 8 . APIs chính 43](#_Toc8306)

[Bảng 9 . Số lượng lỗi được xử lý bởi MyChecker 49](#_Toc10282)

**Mục lục**

[Lời cảm ơn 1](#_Toc32222)

[Tóm tắt 2](#_Toc7960)

[Lời cam đoan 4](#_Toc5486)

[Danh mục từ viết tắt 5](#_Toc6034)

[Danh sách hình vẽ 6](#_Toc18463)

[Danh sách bảng 7](#_Toc7734)

[Mở đầu 10](#_Toc32425)

[Chương 1 Kiến thức nền tảng 12](#_Toc10077)

[1.1 Kiểm thử trong vòng đời phát triển phần mềm 12](#_Toc14477)

[1.1.1 Tổng quan 12](#_Toc26308)

[1.1.2 Vai trò 13](#_Toc27574)

[1.2 Kiểm thử khả năng tiếp cận 15](#_Toc20301)

[1.2.1 Khái niệm 15](#_Toc10674)

[1.2.2 Khuyết tật và khả năng tiếp cận 16](#_Toc13748)

[1.2.3 Các tiêu chuẩn trong kiểm thử khả năng tiếp cận 16](#_Toc14)

[1.2.3.1 Nguyên tắc hỗ trợ tiếp cận nội dung web (WCAG) 2.0 17](#_Toc22025)

[1.2.3.2 Mục 508 (Section 508) 19](#_Toc29805)

[Chương 2 Vấn đề về khả năng tiếp cận của các trang web 21](#_Toc14835)

[2.1 Một số vấn đề phổ biến 21](#_Toc16957)

[2.1.1 Độ tương phản của văn bản 21](#_Toc11777)

[2.1.2 Cấu trúc tiêu đề 21](#_Toc28477)

[2.1.3 Khả năng tiếp cận bàn phím 22](#_Toc9883)

[2.1.4 Liên kết điều hướng 22](#_Toc13632)

[2.1.5 Nội dung không phải HTML 22](#_Toc14663)

[2.1.6 Lỗi văn bản thay thế hình ảnh 22](#_Toc29372)

[2.1.7 Thiếu văn bản liên kết 23](#_Toc13673)

[2.1.8 Các biểu mẫu có cấu trúc kém 23](#_Toc8497)

[2.1.9 Biểu mẫu hết hạn 24](#_Toc32673)

[2.2 Một số công cụ kiểm thử hiện tại 24](#_Toc1898)

[2.2.1 Wave 24](#_Toc11633)

[2.2.2 Taw 25](#_Toc20262)

[2.2.3 Achecker 25](#_Toc2780)

[Chương 3 Xây dựng công cụ hỗ trợ kiểm thử khả năng tiếp cận trang web 26](#_Toc8364)

[3.1 Vấn đề của các công cụ hiện tại 27](#_Toc16912)

[3.2 Phân tích công cụ kiểm thử AChecker 30](#_Toc15446)

[3.3 Xây dựng công cụ cải tiến AChecker 31](#_Toc23907)

[Chương 4 Triển khai và đánh giá công cụ 40](#_Toc2325)

[1.1 Tiến hành xây dựng công cụ 40](#_Toc24099)

[1.1.1 Backend 40](#_Toc2173)

[1.1.2 Fontend 44](#_Toc14096)

[1.2 Thử nghiệm công cụ 47](#_Toc25571)

[1.3 Đánh giá công cụ 49](#_Toc12150)

[Kết luận 51](#_Toc9090)

# Mở đầu

Trong những năm gần, chuyển đổi kỹ thuật số và sự phát triển công nghệ mang lại lợi ích cho mọi người và cải thiện chất lượng cuộc sống của họ thông qua khả năng tiếp cận thông tin. Các ứng dụng web đóng vai trò quan trọng trong việc truyền tải thông tin tới mọi người một cách nhanh chóng. Do đó sự phát triển của ứng dụng web ngày càng nhận được sự quan tâm của cộng đồng. Tuy nhiên, các ứng dụng web hiện nay chưa thực sự đáp ứng được các tiêu chuẩn về khả năng tiếp cận, vẫn còn tồn tại nhiều lỗi tiếp cận ứng dụng web đối với người dùng. Các nghiên cứu gần đây cho thấy nhiều trang web vẫn không thể truy cập được đối với người khuyết tật [1, 2]. Điều này khiến một thị phần lớn thị trường người khuyết tật gặp khó khăn trong việc tiếp cận các ứng dụng web, theo WHO thị phần người khuyết tật chiếm 15% dân số thế giới.

Vì những tác hại của sự thiếu khả năng tiếp cận ứng dụng web, việc phát hiện và phân tích các lỗi trong khả năng tiếp cận ứng dụng web được xem là một yếu tố quan trọng nhằm giúp cho các ứng dụng web ngày càng phát triển và có thể được tiếp cận bởi nhiều người dùng hơn. Phát hiện ra các lỗi của khả năng tiếp cận ứng dụng web nhằm khắc phục sớm trong quá trình phát triển để đáp ứng được các tiêu chuẩn về khả năng tiếp cận của ứng dụng web trước khi được triển khai. Trong những năm gần đây, các công cụ phân tích và tìm ra lỗi về khả năng tiếp cận của ứng dụng web đã được phát triển. Tuy nhiên, các công cụ này chỉ giúp phát hiện các lỗi tiếp cận ứng dụng web mà chưa đưa ra các gợi ý để sửa lỗi. Điều này khiến cho các nhà phát triển phải bỏ nhiều thời gian, công sức và gặp khó khăn trong việc khắc phục lại các lỗi tiếp cận của ứng dụng web. Việc xây dựng một công cụ kiểm thử khả năng tiếp cận ứng dụng web có chức năng gợi ý sửa lỗi sẽ đem lại hiệu quả tốt hơn.

Khóa luận này đề xuất bổ sung chức năng sửa lỗi cho công cụ kiểm thử khả năng tiếp cận ứng dụng web nhằm khắc phục lỗi nhanh chóng và hiệu quả hơn. Công cụ kiểm thử khả năng tiếp cận ứng dụng web mà khóa luận này xây dựng không chỉ cung cấp cho các lập trình viên thông tin chi tiết về các lỗi mà còn đưa ra cách sửa lỗi một cách trực quan, điều này giúp nhà phát triển, kể cả những người không có kiến thức về khả năng tiếp cận ứng dụng web có thể hiểu được và khắc phục các lỗi kịp thời.

Nội dung của khóa luận gồm 4 chương chính. Chương 1 trình bày các kiến thức tổng quan về kiểm thử khả năng tiếp cận ứng dụng web và các tiêu chuẩn trong kiểm thử khả năng tiếp cận ứng dụng web. Một số vấn đề phổ biến về khả năng tiếp cận ứng dụng web và các công cụ kiểm thử hiện tại được trình bày trong chương 2. Từ các kiến thức nền tảng của chương 1 kết hợp với kết quả nghiên cứu về các lỗi vi phạm khả năng truy cập ở chương 2 làm cơ sở lý thuyết giúp chương 3 phân tích các điểm yếu của các công cụ hiện tại và trình bày phương pháp cải tiến công cụ kiểm thử khả năng tiếp cận ứng dụng web nhằm mục đích cải thiện các công cụ kiểm thử hiện có. Cuối cùng, chương 4 sẽ trình bày về cách xây dựng công cụ, những thử nghiệm và đưa ra những đánh giá về công cụ cải tiến mà khóa luận này xây dựng.

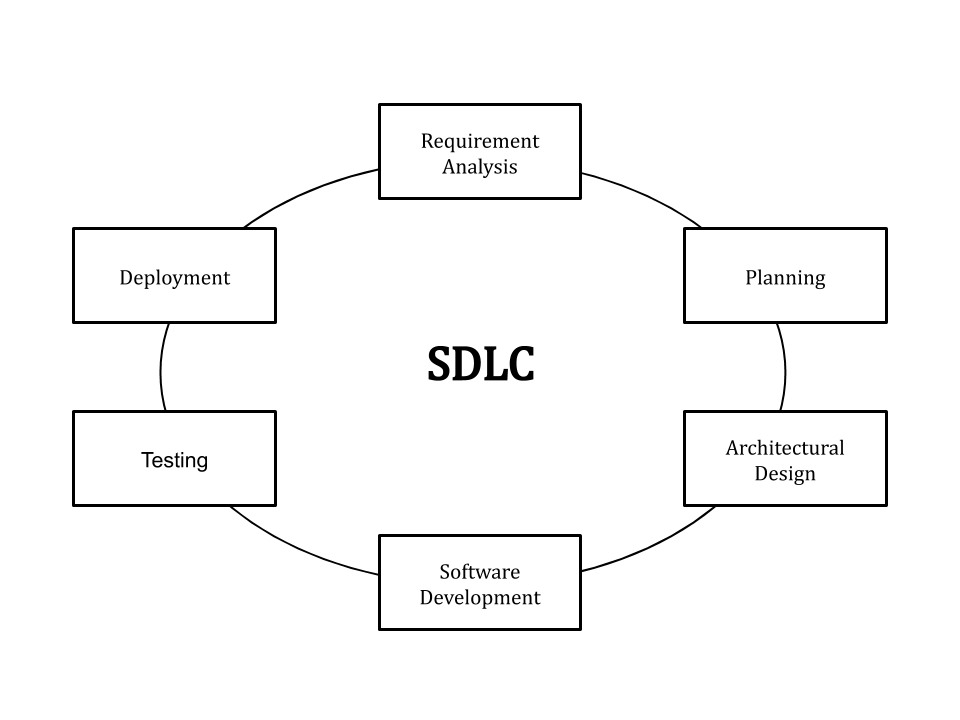
# Chương 1 Kiến thức nền tảng

Chương này sẽ trình bày các kiến thức tổng quan về kiểm thử trong vòng đời phát triển phần mềm, kiểm thử khả năng tiếp cận và những tiêu chuẩn trong kiểm thử khả năng tiếp cận ứng dụng web. Tổng quan về quy trình kiểm thử không chỉ nêu ra những kiến thức nền tảng, đặc điểm mà còn trình bày về tầm quan trọng của nó trong quy trình phát triển phần mềm. Kiểm thử khả năng tiếp cận sẽ trình bày những nguyên lý, tiêu chuẩn về khả năng truy cập ứng dụng web, các ứng dụng web khi được phát triển phải tuân thủ theo những tiêu chuẩn này để hỗ trợ người dùng tiếp cận một cách dễ dàng hơn.

## Kiểm thử trong vòng đời phát triển phần mềm

### Tổng quan

Vòng đời phát triển phần mềm (SDLC) là một quy trình có cấu trúc cho phép sản xuất phần mềm với chất lượng cao, chi phí thấp và trong thời gian sản xuất ngắn nhất có thể. Mục tiêu của SDLC là sản xuất phần mềm ưu việt, đáp ứng thậm chí vượt quá mọi mong đợi và nhu cầu của khách hàng. SDLC xác định và vạch ra một kế hoạch chi tiết với các khâu hoặc các giai đoạn, mỗi giai đoạn bao gồm quy trình và sản phẩm của riêng nó. Việc tuân thủ SDLC nâng cao tốc độ phát triển và giảm thiểu rủi ro và chi phí của các dự án phát triển phần mềm [3]. Một vòng đời phát triển phần mềm bao gồm 6 giai đoạn: phân tích yêu cầu, lên kế hoạch, thiết kế, phát triển, kiểm thử và cuối cùng là triển khai. Mỗi giai đoạn này đều rất quan trọng để đáp ứng các yêu cầu của khách hàng và đạt được các mục tiêu của dự án. Hình 1 dưới đây biểu diễn 6 giai đoạn trong vòng đời phát triển phần mềm.



Hình 1. Vòng đời phát triển phần mềm

Như chúng ta đã biết vòng đời phát triển phần mềm là một quá trình liên tục, do đó giai đoạn kiểm thử là một trong những giai đoạn quan trọng nhất của SDLC để đảm bảo chất lượng sản phẩm ở mỗi chu kỳ phát triển. Nó giúp các công ty thực hiện đánh giá toàn diện về phần mềm và đảm bảo rằng sản phẩm của họ đáp ứng nhu cầu của khách hàng. Các giai đoạn thử nghiệm của vòng đời phát triển phần mềm giúp các công ty xác định tất cả lỗi xảy ra trong phần mềm trước khi giai đoạn triển khai bắt đầu. Nếu lỗi xảy ra trong phần mềm không được giải quyết trước khi triển khai, chúng có thể ảnh hưởng xấu đến hoạt động kinh doanh của khách hàng. Bên cạnh đó, việc cố gắng khắc phục những vấn đề này ở giai đoạn sau có thể phát sinh chi phí đáng kể. Khi các vấn đề tồn tại trong phần mềm bị chậm trễ trong việc phát hiện sẽ dẫn tới chi phí mà công ty có thể phải đối mặt càng lớn.

### Vai trò

Kiểm thử phần mềm là rất quan trọng vì nếu có bất kỳ lỗi nào tiềm ẩn trong phần mềm, nó có thể được xác định sớm và có thể được giải quyết trước khi giao sản phẩm phần mềm tới khách hàng. Sản phẩm phần mềm được kiểm thử đảm bảo độ tin cậy, bảo mật và hiệu suất cao, giúp tiết kiệm thời gian, chi phí, đạt được hiệu quả và sự hài lòng của khách hàng [4]. Một số ý dưới đây cho thấy tầm quan trọng của giai đoạn kiểm thử trong quá trình phát triển phần mềm.

**Nâng cao chất lượng**

Kiểm thử và chất lượng sản phẩm phần mềm đi đôi với nhau. Chất lượng phần mềm có thể được đo lường bằng số lượng lỗi được xác định trong quá trình thử nghiệm và những lỗi đó sẽ được sửa trong vòng đời phát triển phần mềm. Điều này sẽ được thực hiện liên tục trong suốt giai đoạn phát triển, từ đó giúp cải thiện chất lượng của phần mềm. Kiểm thử cho phép nhà phát triển biết được phần mềm hoạt động ra sao từ đó giúp doanh nghiệp có thể đo lường chất lượng của phần mềm trước khi nó đi vào hoạt động.

**Sai sót trong phát triển**

Không có sản phẩm phần mềm nào là hoàn hảo cả và con người khi phát triển đều có thể mắc sai sót, đặc biệt là khi xây dựng một hệ thống phức tạp. Việc phát triển phần mềm mà không kiểm thử không khác nào một trò chơi phỏng đoán. Do đó khi các chức năng được phát triển, điều quan trọng là phải kiểm thử và xác minh rằng các chức năng đó và hệ thống hoạt động như mong đợi theo các yêu cầu hay không.

**Giảm thiểu rủi ro**

Các lỗi phải được tìm ra và sửa chữa hoặc loại bỏ khỏi sản phẩm phần mềm để đảm bảo rằng nó có thể chạy như mong đợi trong quá trình vận hành. Khi một lỗi nghiêm trọng được phát hiện trong khi sản phẩm đã được triển khai thì những ảnh hưởng và mức độ nghiêm trọng cao vì nó ảnh hưởng đến người dùng cuối, vì vậy việc kiểm thử liên tục là rất quan trọng để giảm thiểu rủi ro và đảm bảo rằng phần mềm đã sẵn sàng đưa vào triển khai.

**Giảm thiểu chi phí**

Khi lập kế hoạch kinh phí cho một dự án phần mềm, điều mong muốn của các nhà quản lý dự án là kinh phí triển khai dự án sẽ không vượt quá kinh phí đã đề ra. Tuy nhiên các dự án thường có cảm giác như thể ngân sách sẽ cạn kiệt sớm hơn dự kiến, hoặc chi phí cao hơn vào cuối dự án. Tại sao rất nhiều tiền được chi vào việc khắc phục các sự cố hệ thống? Việc sửa chữa một lỗi thường tốn nhiều thời gian và khó khăn hơn, do đó tốn kém hơn, khi lỗi đó được phát hiện ở giai đoạn sau khi mà một lượng lớn các chức năng đã được xây dựng. Do đó việc kiểm thử sớm và liên tục không chỉ giúp sản phẩm phần mềm tốt hơn mà còn cho phép người quản lý dự án kiểm soát tốt hơn ngân sách và thời gian. Việc phát hiện ra các lỗi và sửa chữa chúng ở giai đoạn sớm hơn có thể giảm chi phí không mong muốn khi kết thúc dự án.

**Giữ niềm tin khách hàng**

Cuối cùng, tất cả những điều trên đều dẫn đến mục đích là sự hài lòng của khách hàng. Sự ổn định của sản phẩm phần mềm là điều mà khách hàng mong muốn. Kiểm thử phần mềm trong suốt vòng đời phát triển cung cấp thông tin về phần mềm đã được phát triển tốt và ổn định như thế nào, do đó làm tăng niềm tin ở khách hàng sau khi sản phẩm được đưa vào triển khai tới người dùng.

## Kiểm thử khả năng tiếp cận

### Khái niệm

Kiểm thử khả năng tiếp cận là một phần trong quy trình kiểm thử phần mềm. Nó được áp dụng để kiểm tra tính dễ sử dụng của một trang web đối với người dùng. Loại kiểm thử này sử dụng các công cụ hỗ trợ nhằm thể đánh giá nội dung trang web và xác định xem nội dung đó có thể truy cập được đối với người bị hạn chế về khả năng sử dụng hay không. Mục tiêu của kiểm thử khả năng tiếp cận là làm cho việc tiệp cận thông tin trên Internet trở nên dễ dàng hơn đối với mọi người. Tiêu chí thành công của kiểm thử khả năng tiếp cận được sắp xếp dựa trên bốn nguyên tắc về khả năng tiếp cận [5] như sau:

**Khả năng nhận biết**

Thông tin và các thành phần giao diện người dùng phải hiển thị cho người dùng theo cách họ có thể nhận thức được. Người dùng phải có khả năng tiếp nhận thông tin được trình bày thông qua ít nhất một trong các giác quan của họ.

**Khả năng hoạt động**

Các thành phần giao diện người dùng và điều hướng phải có thể hoạt động được một cách linh hoạt. Qua đó người dùng có khả năng tương tác với giao diện một cách thuận lợi và dễ dàng hơn.

**Tính dễ hiểu**

Thông tin và các hoạt động tương tác trên giao diện người dùng phải dễ hiểu. Từ đó giúp người dùng có thể hiểu được thông tin cũng như hoạt động của giao diện người dùng.

**Tính mạnh mẽ**

Nội dung của các ứng dụng web phải đủ rành mạch, rõ ràng, dễ hiểu để có thể diễn giải một cách đáng tin cậy tới người dùng, các nội dung này cũng phải có thể được hiểu bởi các công nghệ hỗ trợ. Ngoài ra khi công nghệ hoặc các nền tảng trình duyệt phát triển, nội dung vẫn có thể truy cập được bởi người dùng.

### Khuyết tật và khả năng tiếp cận

Theo Tổ chức Y tế Thế giới (WHO), khuyết tật là một phần của tình trạng con người. Khuyết tật là kết quả của sự tương tác giữa những người với những điều kiện nhất định và những rào cản về môi trường, điều này cản trở sự tham gia của họ vào xã hội trên cơ sở bình đẳng với những người khác. Do đó, khuyết tật không phải là thuộc tính của con người mà nó phụ thuộc vào những rào cản mà người khuyết tật gặp phải trong cuộc sống hàng ngày. Hiện nay, hơn một tỷ người đang sống với một số dạng khuyết tật. Nó chiếm khoảng 15% tổng dân số thế giới [6], hơn nữa số lượng người khuyết tật ngày càng tăng lên đáng kể khi tính đến không chỉ khuyết tật vĩnh viễn mà còn cả những người khuyết tật tạm thời do bệnh tật hoặc tai nạn. Khuyết tật được chia làm hai dạng đó là khuyết tật cá nhân và khuyết tật phi cá nhân. Thứ nhất, khuyết tật cá nhân là những khuyết tật liên quan đến những khiếm khuyết về cơ thể hoặc tinh thần của con người có thể do bẩm sinh hoặc mắc phải tại bất kỳ thời điểm nào trong cuộc đời của một người, ví dụ: thị giác, thính giác, lời nói, vận động, nhận thức và tâm lý xã hội. Thứ hai, khuyết tật phi cá nhân là những khuyết tật liên quan đến các tình huống trong môi trường xung quanh con người có thể xảy ra tại bất kỳ thời điểm nào trong cuộc sống của một người và thường là tạm thời, ví dụ: các vấn đề về nhận thức do các rào cản về ngôn ngữ, tôn giáo hoặc văn hóa, điều kiện môi trường, tính khả dụng của công nghệ. Môi trường của một người có tác động rất lớn đến mức độ khuyết tật của họ. Môi trường không thể tiếp cận tạo ra khuyết tật bằng cách tạo ra các rào cản. Ngược lại, các môi trường dễ tiếp cận làm giảm thiểu tình trạng khuyết tật và tạo điều kiện cho sự tham gia và hòa nhập với cộng đồng một cách dễ dàng hơn. Tổ chức Tiêu chuẩn hóa Quốc tế (ISO) định nghĩa khả năng tiếp cận là “khả năng sử dụng của một sản phẩm, dịch vụ, môi trường hoặc cơ sở vật chất bởi người sử dụng với phạm vi về khả năng sử dụng rộng nhất” [7]

### 1.2.3 Các tiêu chuẩn trong kiểm thử khả năng tiếp cận

Có hai tiêu chuẩn về khả năng tiếp cận ứng dụng web hiện đang được sử dụng phổ biến trên thế giới đó là: nguyên tắc hỗ trợ tiếp cận nội dung web (WCAG) 2.0 [8] và mục 508 (Section 508) [9] của Đạo luật Phục hồi Hoa Kỳ.

Tiêu chuẩn WCAG do W3C phát triển, kể từ khi được thành lập vào năm 1994, W3C đã khởi xướng các giao thức chung cho sự phát triển của ứng dụng web với một sự nỗ lực được gọi là Sáng kiến hỗ trợ truy cập web (WAI), bằng những nỗ lực đó sau này đã tạo ra WCAG. WCAG 1.0 [10] được phát triển vào cuối những năm 1990 và đã được hoàn thiện vào năm 1999 trước khi WCAG bản 2.0 được đề xuất vào năm 2000 và trở thành chính thức vào năm 2008 bởi W3C. Kể từ đó WCAG đã được sử dụng rộng rãi trong cả về hướng dẫn thiết kế ứng dụng web và kiểm thử, đánh giá trang web và nó cũng được chấp nhận như một tiêu chuẩn về khả năng truy cập web quốc tế. Từ đó với việc các ứng dụng web phải tuân thủ theo tiêu chuẩn WCAG này tạo điều kiện thuận lợi cho người khuyết tật truy cập web một cách dễ dàng hơn. Theo định nghĩa của tiêu chuẩn này, “hướng dẫn về những nguyên tắc hỗ trợ khả năng tiếp cận ứng dụng cung cấp một nền tảng làm cho nội dung web dễ tiếp cận hơn với người khuyết tật”. Khả năng tiếp cận liên quan đến một loạt các loại khuyết tật khác nhau, bao gồm khuyết tật về thị giác, thính giác, thể chất, lời nói, nhận thức, ngôn ngữ và thần kinh. Mặc dù các hướng dẫn về khả năng tiếp cận này có thể khắc phục nhiều vấn đề, nhưng chúng không thể giải quyết được hết mọi nhu cầu của người sử dụng với nhiều dạng, nhiều mức độ và sự kết hợp của các loại khuyết tật. Những hướng dẫn này có thể làm cho nội dung web dễ sử dụng hơn đối với những người lớn tuổi với khả năng sử dụng của họ thay đổi do lão hóa nói riêng và cải thiện khả năng sử dụng cho mọi người dùng nói chung.

Đối với mục 508 (Section 508), đó là một đạo luật được sửa đổi vào năm 1998 của Đạo luật Phục hồi của Hoa Kỳ năm 1973, trong đó nó yêu cầu các cơ quan Liên bang cung cấp khả năng truy cập công nghệ thông tin và điện tử (EIT) cho người khuyết tật. Theo luật, tất cả các cơ quan Liên bang phải cung cấp cho nhân viên khuyết tật và các thành viên của công chúng khả năng truy cập thông tin tương tự với những người bình thường khác.

#### 1.2.3.1 Nguyên tắc hỗ trợ tiếp cận nội dung web (WCAG) 2.0

Như đã nêu ra bởi W3C, WCAG 2.0 bao gồm một loạt các khuyến nghị để làm cho nội dung Web dễ tiếp cận hơn, điều này sẽ làm cho nội dung dễ tiếp cận và dễ sử dụng hơn đối với người dùng, đặc biệt là những người khuyết tật về thị giác, thính giác, thể chất, lời nói, nhận thức, ngôn ngữ, thần kinh, nhạy cảm với ánh sáng và cả sự kết hợp giữa các loại khuyết tật này. Do đó để cải thiện khả năng truy cập của các ứng dụng, WCAG 2.0 đã đề xuất 4 nguyên tắc với 12 lớp hướng dẫn để các ứng dụng web tuân thủ như mô tả trong bảng 1 dưới đây.

Bảng 1. Nguyên tắc hỗ trợ tiếp cận nội dung web

|  |  |
| --- | --- |
| **Nguyên tắc** | **Nội dung** |
| Nguyên tắc 1 - Có thể nhận thức được: Thông tin và các thành phần giao diện người dùng phải hiển thị cho người dùng theo cách họ có thể nhận thức được. | 1. Các lựa chọn thay thế văn bản: Cung cấp các lựa chọn thay thế văn bản cho bất kỳ nội dung không phải văn bản.  2. Phương tiện dựa trên thời gian: Cung cấp các lựa chọn thay thế cho phương tiện dựa trên thời gian.  3. Thích ứng: Tạo nội dung có thể được trình bày theo nhiều cách khác nhau (ví dụ bố cục đơn giản hơn) mà không làm mất thông tin hoặc cấu trúc.  4. Dễ phân biệt: Giúp người dùng nhìn và nghe nội dung dễ dàng hơn bao gồm cả việc tách tiền cảnh khỏi hậu cảnh. |
| Nguyên tắc 2 - Có thể hoạt động: Các thành phần giao diện người dùng và điều hướng phải có khả năng hoạt động. | 1. Bàn phím có thể truy cập: Cung cấp tất cả các chức năng từ bàn phím.  2. Đủ thời gian: Cung cấp cho người dùng đủ thời gian để đọc và sử dụng nội dung.  3. Thiết kế: Không thiết kế nội dung theo cách được biết là có thể gây ra rối mắt.  4. Điều hướng: Cung cấp các cách giúp người dùng điều hướng, tìm nội dung và xác định vị trí của họ. |
| Nguyên tắc 3 - Dễ hiểu: Thông tin và hoạt động của giao diện người dùng phải dễ hiểu. | 1. Có thể đọc được: Làm cho nội dung văn bản dễ đọc và dễ hiểu.  2. Dự đoán: Làm cho các trang web xuất hiện và hoạt động theo những cách có thể dự đoán được.  3. Gợi ý đầu vào: Giúp người dùng tránh lỗi và sửa lỗi. |
| Nguyên tắc 4 - Mạnh mẽ: Nội dung phải rõ ràng để có thể diễn giải một cách đáng tin cậy tới người dùng, và được hiểu bởi các công nghệ hỗ trợ. | 1. Tương thích: Tối đa hóa khả năng tương thích với các tác nhân người dùng, bao gồm cả các công nghệ hỗ trợ. |

Ngoài ra, WCAG 2.0 cũng thiết lập ba cấp độ yêu cầu tuân thủ là A (thấp nhất), AA và AAA (cao nhất). Cấp độ A - mức độ tuân thủ thấp nhất đã chỉ định các yếu tố bắt buộc của ứng dụng web để người khuyết tật có thể truy cập thông tin trên các ứng dụng. Trong khi đó, cấp độ AA liệt kê các yêu cầu nâng cao dường như sẽ loại bỏ các rào cản về khả năng truy cập để một nhóm đối tượng người dùng rộng hơn có thể truy cập nội dung web. Mức tuân thủ cao nhất - mức AAA quy định các tính năng nâng cao khác đảm bảo khả năng truy cập web tới phạm vi người dùng rộng rãi nhất có thể. Theo định nghĩa trong WCAG, ít nhất một trong các mức yêu cầu được đề cập, đặc biệt là mức A phải được đáp ứng đầy đủ cho bất kỳ trang web nào tuân thủ theo tiêu chuẩn WCAG.

#### 1.2.3.2 Mục 508 (Section 508)

Mục 508 của Đạo luật phục hồi của Hoa Kỳ chỉ ra 16 tiêu chuẩn để tuân thủ khả năng truy cập web để đảm bảo nội dung web dễ truy cập hơn. Các tiêu chuẩn này được thể hiện trong bảng 2 dưới đây.

Bảng 2. Tiêu chuẩn theo mục 508

|  |  |
| --- | --- |
| **Tiêu chuẩn** | **Nội dung** |
| Văn bản tương đương | Đối với mọi phần tử không phải văn bản sẽ được cung cấp văn bản thay thế(ví dụ: thông qua "alt", "long desc" hoặc trong nội dung phần tử). |
| Đồng bộ hóa tương đương | Các lựa chọn thay thế tương đương cho bất kỳ bản trình bày đa phương tiện nào sẽ được đồng bộ với bản trình bày. |
| Màu sắc | Các trang web phải được thiết kế sao cho tất cả thông tin được truyền tải có màu sắc. |
| Kiểu định dạng | Các tài liệu phải được sắp xếp để chúng có thể đọc được mà không cần phải có một kiểu định dạng đặc biệt. |
| Liên kết văn bản cho hình ảnh | Các liên kết văn bản dự phòng sẽ được cung cấp cho từng hình ảnh. |
| Bản đồ hình ảnh phía máy khách thay vì phía máy chủ | Bản đồ hình ảnh phía máy khách sẽ được cung cấp thay vì bản đồ hình ảnh phía máy chủ ngoại trừ trường hợp không thể xác định các vùng bằng hình dạng hình học có sẵn. |
| Tiêu đề hàng/ cột cho bảng dữ liệu | Tiêu đề hàng và cột phải được xác định cho các bảng dữ liệu. |
| Liên kết các ô dữ liệu và ô tiêu đề | Đánh dấu sẽ được sử dụng để liên kết các ô dữ liệu và ô tiêu đề cho các bảng dữ liệu có hai hoặc nhiều mức logic của tiêu đề hàng hoặc cột. |
| Khung sẽ có tiêu đề | Khung phải được đặt tiêu đề bằng văn bản tạo điều kiện thuận lợi cho việc nhận dạng và điều hướng khung. |
| Tránh nhấp nháy | Các trang phải được thiết kế để tránh làm cho màn hình nhấp nháy với tần số lớn hơn 2 Hz và thấp hơn 55Hz |
| Trang văn bản | Một trang chỉ văn bản, với thông tin hoặc chức năng tương đương, sẽ được cung cấp để làm cho một trang web tuân thủ các quy định. |
| Tập lệnh phải có văn bản chức năng | Thông tin do tập lệnh cung cấp sẽ được nhận dạng bằng văn bản chức năng có thể đọc được bằng công nghệ hỗ trợ. |
| Applet, v.v. phải tuân thủ | Một liên kết được cung cấp đến một trang nơi plugin có thể được tải xuống. |
| Các hình thức có thể truy cập | Biểu mẫu sẽ cho phép những người sử dụng công nghệ hỗ trợ truy cập thông tin, các trường thông tin và chức năng cần thiết. |
| Bỏ qua các liên kết điều hướng lặp lại | Một phương pháp sẽ được cung cấp cho phép người dùng bỏ qua các liên kết điều hướng lặp đi lặp lại. |
| Phản hồi theo thời gian | Khi cần phản hồi theo thời gian, người dùng sẽ được cảnh báo thời gian còn lại để xử lý. |

# Chương 2 Vấn đề về khả năng tiếp cận của các ứng dụng web

Trong chương này sẽ đưa ra một số vấn đề phổ biến về khả năng tiếp cận của các ứng dụng web hiện nay. Và đặc biệt là những phân tích về các công cụ kiểm thử khả năng tiếp cận hiện có. Khóa luận sẽ tiến hành tìm hiểu về 3 công cụ sau: Achecker, Taw và Wave. Mỗi công cụ sẽ giới cụ thể về đặc điểm và cách hoạt động của nó.

## 2.1 Một số vấn đề phổ biến

### 2.1.1 Độ tương phản của văn bản

Một trong những mối quan tâm về khả năng tiếp cận phổ biến nhất là độ tương phản màu sắc thấp giữa văn bản và nền. Độ tương phản màu sắc là một yếu tố thiết yếu của khả năng truy cập web vì nó giúp người dùng cảm nhận nội dung trên một trang web. Theo WCAG, độ tương phản là sự khác biệt về độ sáng giữa hai màu. Độ tương phản thấp trên văn bản là một vấn đề phổ biến về khả năng tiếp cận, vì nhiều người sử dụng có độ nhạy tương phản thấp, chẳng hạn như người lớn tuổi hoặc những người bị mù màu có thể không phân biệt được văn bản và nền của một số màu có tỷ lệ tương phản thấp [11].

Vào tháng 2 năm 2021, WebAIM đã tiến hành đánh giá khả năng truy cập của một triệu trang web và nhận thấy rằng 86,4% trang web đó có văn bản với độ tương phản thấp [12]. Theo WCAG 2.0 mức AA, tỷ lệ tương phản màu lý tưởng là 7: 1 cho văn bản bình thường và 4,5: 1 cho văn bản lớn.

### 2.1.2 Cấu trúc tiêu đề

Người dùng trang web bị khiếm thị hiếm khi đọc toàn bộ trang web. Thay vào đó, họ quét trang để tìm những phần mà họ quan tâm nhất. Họ cũng có thể quét các trang web để tìm thông tin thích hợp bằng cách sử dụng trình đọc màn hình của họ để liệt kê các tiêu đề. Nếu các tiêu đề không được sử dụng đúng cách thì người dùng sẽ không thể tìm thấy những gì họ đang tìm kiếm trên trang. Đó là lý do tại sao ứng dụng web không bao giờ nên sử dụng các tiêu đề theo một cách trang trí và nên đặt chúng theo cách giảm dần hợp lý trên trang (H1, H2, H3, v.v.).

### 2.1.3 Khả năng tiếp cận bàn phím

Sử dụng chuột để điều hướng một trang web có thể khó hoặc thậm chí là không thể đối với những người bị khiếm thị, vì vậy thay vào đó, họ sử dụng bàn phím để điều hướng. Những người khiếm thị có thể di chuyển qua một trang web bằng các lệnh bàn phím và phím tắt. Tuy nhiên, một trang web sẽ phải có nội dung có cấu trúc tốt với cấu trúc tiêu đề thích hợp và được mã hóa để cho phép bàn phím truy cập vào tất cả nội dung và chức năng như siêu liên kết, liên kết, biểu mẫu và điều khiển phương tiện.

Người dùng bàn phím sử dụng phím Tab để điều hướng trang và khi họ làm vậy, cần có chỉ báo tiêu điểm bàn phím hiển thị cho họ biết họ đang ở thành phần nào trên trang. Các chỉ báo này xuất hiện dưới dạng đường viền hoặc điểm nhấn xung quanh phần tử được lấy nét, giúp những người có thị lực kém xác định vị trí của họ trên trang web. Điều quan trọng là trang web được mã hóa để điều hướng bằng bàn phím vì chỉ báo tiêu điểm cần tuân theo thứ tự đọc hợp lý.

### 2.1.4 Liên kết điều hướng

Các liên kết điều hướng cũng là một vấn đề về khả năng truy cập. Trình đọc màn hình không thể phát hiện điều hướng nếu không được mã hóa tốt. Và khi nó được tìm thấy trên một trang, trình đọc màn hình sẽ không thể bỏ qua nó. Điều này có nghĩa là người dùng bị khiếm thị sẽ phải nghe điều hướng mỗi khi họ tải một trang mới. Một giải pháp cho điều này là chỉ định chính xác các vai trò của các thuộc tính ARIA cho các menu điều hướng để chỉ ra mục đích của nó và giúp nó có thể điều hướng được. Ngoài ra, hãy đảm bảo rằng liên kết chuyển đến nội dung chính có sẵn để người dùng trình đọc màn hình có thể bỏ qua điều hướng.

### 2.1.5 Nội dung không phải HTML

Các vấn đề về khả năng truy cập cũng đề cập đến nội dung trên trang web như tài liệu được lưu trữ trên một trang web. Nội dung không phải HTML, chẳng hạn như PDF, bản trình bày PowerPoint hoặc tài liệu Word, cũng phải được cung cấp khả năng truy cập càng nhiều càng tốt. Có nhiều cách để làm cho các loại tài liệu này có thể truy cập được: với tài liệu PDF, nó có thể được gắn thẻ để điều hướng bằng trình đọc màn hình, trong khi PowerPoint và Word, có thể cung cấp chức năng để tạo tài liệu có thể truy cập.

### 2.1.6 Lỗi văn bản thay thế hình ảnh

Tạo một trang web có thể truy cập không có nghĩa là bạn có thể bỏ qua hình ảnh. Không bao gồm văn bản thay thế cho hình ảnh dẫn đến những trải nghiệm tiêu cực đối với người sử dụng và khiến cho việc hiểu nội dung web khi sử dụng trình đọc màn hình trở nên khó khăn hơn. Văn bản thay thế được tạo ra phải mô tả ngắn gọn, súc tích và rõ ràng thông tin mà hình ảnh truyền đạt vì người dùng trình đọc màn hình có xu hướng điều hướng nhanh chóng qua nội dung. Đối với hình ảnh trang trí, các văn bản thay thế nên được để trống. Thao tác này sẽ giúp trình đọc màn hình bỏ qua hình ảnh không có ý nghĩa đó.

Lưu ý rằng việc thiếu văn bản thay thế cũng ảnh hưởng tiêu cực đến trang web theo những cách khác nhau, chẳng hạn như giảm điểm xếp hạng tìm kiếm trên internet. Nếu hình ảnh thiếu văn bản thay thế hoặc thuộc tính thuộc tính của văn bản thay thế bị bỏ trống, trình đọc màn hình sẽ vẫn thông báo hình ảnh mà không có bất kỳ mô tả nào, điều này có thể khiến người dùng rất khó hiểu.

### 2.1.7 Thiếu văn bản liên kết

Các liên kết rất quan trọng trong việc điều hướng web, nhưng chúng lại gây ra một loạt các vấn đề về khả năng truy cập của riêng chúng. Thiếu văn bản liên kết là một trở ngại phổ biến đối với những người khiếm thị. Điều này xảy ra khi không có văn bản nào được sử dụng để mô tả hoặc trình bày rõ ràng về các liên kết đối với các hình ảnh chức năng như nút, biểu trưng, biểu tượng hoặc các hình ảnh khác được sử dụng làm liên kết. Do đó trình đọc màn hình không thể đọc những hình ảnh này hoạt động như các liên kết. Để tránh điều này xảy ra, văn bản thay thế hình ảnh phải truyền đạt mục đích của hình ảnh và hành động sẽ được thực hiện, thay vì chỉ mô tả về hình ảnh.

### 2.1.8 Các biểu mẫu có cấu trúc kém

Hầu hết các trang web đều có các trường biểu mẫu để người dùng điền vào và các trường biểu mẫu này thường gây ra các vấn đề về khả năng truy cập đối với những người khiếm thị. Một trong những vấn đề lớn nhất với biểu mẫu là chúng thường không có nhãn và hướng dẫn chính xác mà trình đọc màn hình cần để truyền tải thông tin về biểu mẫu. Mỗi trường thông tin cần một nhãn có thể truy cập để trình đọc màn hình đọc cho người dùng biết những gì cần điền vào. Các nút cũng cần có văn bản rõ ràng để người dùng biết phải chọn gì sau khi hoàn thành biểu mẫu. Nếu ứng dụng web sử dụng CAPTCHA, thì cần phải có âm thanh hoặc phương tiện thay thế có thể truy cập khác cho người dùng bị khiếm thị.

Ngoài ra, khi biểu mẫu được điền không chính xác thì điều gì sẽ xảy ra. Thông thường, các trang web sẽ trả về biểu mẫu với các trường thông tin không chính xác có màu đỏ. Điều này rõ ràng là một vấn đề về khả năng tiếp cận đối với những người không thể nhìn thấy trường thông tin đó. Thay vào đó, ứng dụng web phải hỗ trợ đọc ra văn bản để giải thích phần nào của biểu mẫu trả lại lỗi.

### 2.1.9 Biểu mẫu hết hạn

Nhiều biểu mẫu có chức năng bảo mật khiến trang web hoặc biểu mẫu hết hạn sau một khoảng thời gian nhất định. Tuy nhiên, đối với những người sử dụng công nghệ hỗ trợ để điền vào các biểu mẫu, quá trình này có thể mất nhiều thời gian hơn và họ có thể không biết rằng có giới hạn thời gian. Do đó ứng dụng web phải cung cấp đầy đủ thông tin rằng biểu mẫu có giới hạn thời gian hoặc cung cấp chức năng cho người dùng công nghệ hỗ trợ để kéo dài thời hạn kết thúc quá trình xử lý biểu mẫu.

## 2.2 Một số công cụ kiểm thử hiện tại

Các ứng dụng web khi phát triển còn tồn tại rất nhiều lỗi vi phạm khả năng tiếp cận. Để giảm thiểu các lỗi vi phạm đó trước khi sản phẩm triển khai tới người dùng, trong những năm gần đây, nhiều công cụ và khuôn mẫu được nghiên cứu và phát triển để phân tích lỗi vi phạm khả năng tiếp cận nhằm phát hiện các lỗi vi phạm sớm giúp cho việc khắc phục lỗi trở lên nhanh chóng và dễ dàng hơn. Dưới đây sẽ cung cấp cái nhìn tổng quan về một số công cụ kiểm thử tiêu biểu.

### 2.2.1 Wave

Wave [13] - Web Accessibility Assessment Tool là một công cụ kiểm thử khả năng truy cập ứng dụng web miễn phí được phát triển bởi WebAIM. Đây là một công cụ kiểm thử sự tuân thủ khả năng truy cập của các ứng dụng web theo các nguyên tắc và các hướng dẫn khả năng tiếp cận khuyến nghị quốc tế như WCAG 2.0 và USA Section 508. Wave cung cấp khả năng kiểm thử cục bộ một trang web thông qua các tiện ích mở rộng có thể được nhúng trong Firefox và Chrome, cho phép các nhà phát triển một cách đơn giản để kiểm tra nội dung web trước khi được triển khai, ngoài ra nó còn cho phép phân tích nhanh chóng một ứng dụng web ngay trên nền tảng web bằng cách cung cấp thông tin URL trang web cần kiểm thử đó.

WAVE hiện thị trực quan kết quả phân tích lỗi trên một bản snapshot của trang web với các lỗi vi phạm được biểu diễn bằng các biểu tượng trên đó. Từ đó phép người dùng và chuyên gia kiểm tra các vấn đề về khả năng truy cập tiềm ẩn. Các biểu tượng màu đỏ đề cập đến lỗi trợ năng, biểu tượng màu vàng biểu thị cảnh báo, biểu tượng màu xanh lá cây biểu thị các tính năng trợ năng và tất cả các biểu tượng màu xanh lam nhạt biểu thị các yếu tố cấu trúc, ngữ nghĩa hoặc điều hướng. Công cụ cũng tạo ra một báo cáo kết quả phân tích chứa số lỗi và các nguy cơ tiềm ẩn có thể xảy. WAVE không có công cụ sửa chữa và không hiển thị các hướng dẫn khả năng truy cập đã triển khai.

### 2.2.2 Taw

TAW [14] là một công cụ kiểm thử miễn phí được phát triển bởi Trung tâm Phát triển Công nghệ Thông tin và Truyền thông Tây Ban Nha ở Asturias (CTIC). Công cụ này có thể được sử dụng như một công cụ trợ năng trực tuyến để kiểm tra các lỗi vi phạm khả năng tiếp cận trên các trang web hoặc như một ứng dụng java độc lập phù hợp với nhiều nền tảng (Microsoft Windows, Mac OSX, Linux, Solaris). Công cụ TAW cũng có sẵn dưới dạng một trình cắm có thể được thêm vào trong các trình duyệt rất tiện lợi. Tiêu chí được áp dụng để phân tích lỗi vi phạm khả năng tiếp cận của các ứng dụng web là các nguyên tắc về khả năng truy cập web của W3C. Công cụ cung cấp các tùy chọn về các khuyến nghị hướng dẫn tiếp cận nội dung Web như WCAG 1.0 hoặc WCAG 2.0. Ngoài ra nó cũng cung cấp tùy chọn để chọn cấp độ phân tích lỗi vi phạm bao gồm 3 cấp độ: cấp độ A, cấp độ AA và cấp độ AAA.

Sau khi phân tích lỗi vi phạm của ứng dụng web, TAW tạo ra một báo cáo tóm tắt dựa trên trang được phân tích với thông tin về kết quả của việc kiểm thử. Trong phần báo cáo, công cụ đưa ra số lượng lỗi được thực hiện phân tích trong nội dung trang web trên. Kết quả phân tích cung cấp một cái nhìn tổng quan dựa trên các nguyên tắc về khả năng tiếp cận trong WCAG. Các vấn đề về khả năng tiếp cận do TAW đưa ra gồm ba loại vấn đề bao gồm: cảnh báo, các lỗi chưa được xem xét, và các lỗi được xác định. TAW không hiển thị các hướng dẫn truy cập ứng dụng web ứng với từng lỗi vi phạm. Ngoài ra, công cụ có thể gửi một báo cáo đầy đủ tới địa chỉ email của người dùng nếu cần thiết. Giao diện công cụ được thiết kế với đa ngôn ngữ giúp người dùng có thể sử dụng dễ dàng hơn.

### 2.2.3 Achecker

Được phát hành lần đầu tiên vào năm 2005, AChecker [15, 16] được tạo ra với mục tiêu cung cấp một công cụ kiểm tra khả năng truy cập 100% minh bạch, có tính tương tác cao, có thể các nhân hóa và miễn phí. Công cụ cung cấp một giải pháp về khả năng tiếp cận cho các ứng dụng web được áp dụng dựa trên tất cả các nguyên tắc hỗ trợ truy cập web được khuyến nghị trên toàn cầu. Achecker là một công cụ trực tuyến để kiểm thử khả năng truy cập của các ứng dụng web nhằm hỗ trợ khả năng truy cập tới những người dùng bị các dạng khuyết tật khác nhau. Nó đánh giá các trang HTML theo các khuyến nghị hướng dẫn khả năng truy cập quốc tế như: BITV 1.0, Mục 508, Stanca Act, WCAG 1.0 và WCAG 2.0.

AChecker đánh giá nội dung HTML cho các vấn đề về khả năng truy cập web bằng cách nhập URL của trang web, tải lên tệp HTML hoặc dán mã nguồn HTML lên công cụ phân tích lỗi. Sau khi phân tích lỗi vi phạm của ứng dụng web, AChecker tạo một báo cáo về tất cả các vấn đề về khả năng truy cập web cho các nguyên tắc đã chọn với ba loại lỗi sau:

* Lỗi đã biết: Đây là những lỗi đã được xác định một cách chắc chắn là các rào cản về khả năng tiếp cận. Các nhà phát triển cần sửa lỗi để khắc phục những sự cố này.
* Lỗi có thể xảy ra: Đây là những lỗi đã được xác định là rào cản có thể xảy ra, nhưng yêu cầu người dùng đưa ra quyết định xác định rằng đó có phải lỗi hay không.
* Lỗi đề tiềm ẩn: Đây là những vấn đề mà AChecker không thể xác định được, cần phải có quyết định của con người. Những lỗi này cần phải được xem xét và khắc phục kịp thời tuy nhiên trong nhiều trường hợp, người dùng có thể xác nhận rằng vấn đề được mô tả không phải không phải là lỗi.

AChecker là công cụ phân tích lỗi truy cập bán tự động vì nó không thể xác minh vấn đề xảy ra trong quá trình phân tích lỗi vi phạm. Giao diện công cụ được thiết kế với nền tảng đa ngôn ngữ giúp người dùng tiếp cận dễ dàng hơn.

# Chương 3 Xây dựng công cụ hỗ trợ kiểm thử khả năng tiếp cận trang web

Chương này tiến hành phân tích các điểm yếu của các công cụ kiểm thử khả năng tiếp cận ứng dụng web hiện tại và trình bày phương pháp cải tiến công cụ, chi tiết về các module xử lý luồng nghiệp vụ để cải thiện các công cụ kiểm thử hiện tại.

## 3.1 Vấn đề của các công cụ hiện tại

Các ứng dụng web cần phải được kiểm thử trước khi triển khai và đi vào hoạt động, việc thực hiện đầy đủ các quy trình kiểm thử nhằm phát hiện ra lỗi có thể xảy ra trong hệ thống, từ đó nhà phát triển có thế khắc phục các lỗi đó một cách kịp thời. Trong đó kiểm thử khả năng tiếp cận ứng dụng web là một trong những khâu quan trọng trong quy trình kiểm thử. Tuy nhiên việc lựa chọn công cụ kiểm thử khả năng tiếp cận ứng dụng web phù hợp và hiệu quả cũng đem đến những sự khó khăn cho các công ty phát triển phần mềm. Tùy vào yêu cầu cũng như ngân sách của các công ty phát triển ứng dụng bên cạnh đó là một vài tiêu chí về các công cụ kiểm thử, từ đó các công ty có thể đưa ra sự lựa chọn công cụ phù hợp nhất.

**Công cụ miễn phí và công cụ thương mại**

Hiện nay trên thị trường có rất nhiều công cụ kiểm thử khả năng tiếp cận ứng dụng web, các công cụ này được chia làm hai loại là các công cụ có thể sử dụng miễn phí và loại thứ hai là các công cụ thương mại, nghĩa là các công ty hoặc cá nhân muốn sử dụng các công cụ đó phải mua bản quyền để có thể sử dụng chúng. Các công cụ miễn phí yêu cầu người dùng cần phải có nhiều kiến thức về tiếp cận ứng dụng web hơn, tuy nhiên việc đào tạo sử dụng các công cụ này lại tốn ít thời gian hơn do các chức năng thường khá đơn giản và dễ sử dụng. Chi phí phải trả để sử dụng công cụ gắn liền với quy mô kiểm thử ứng dụng mà nó đem lại. Các công cụ miễn phí chỉ kiểm tra được một trang web cho mỗi lần thực hiện trong khi đó các công cụ thương mại có thể kiểm tra, phân tích lỗi và thậm chí là cả sửa lỗi cả một ứng dụng web lớn. Việc thu thập thông tin kiểm thử của các ứng dụng web cũng rất quan trọng, nhà phát triển được yêu cầu phải có các báo cáo chi tiết về khả năng truy cập các ứng dụng web một cách cụ thể, và các báo cáo này chỉ được cung cấp bởi các công cụ kiểm thử bản thương mại.

**Nền tảng công công cụ**

Một số công cụ hoạt động trên nền tảng web, vì vậy người dùng không phải tải xuống hoặc cài đặt bất kỳ ứng dụng nào. Với giao diện web thân thiện, người dùng có kiểm tra khả năng tiếp cận dễ dàng và nhanh chóng. Một số công cụ khác hoạt động trong trình duyệt. Chúng là các tiện ích mở rộng được tải xuống trên hầu hết các trình duyệt phổ biến và xuất hiện dưới dạng thanh công cụ hoặc menu bổ sung trong trình duyệt. Có những công cụ được nhúng trực tiếp vào quá trình tạo ứng dụng web, qua đó việc phát triển ứng dụng có thể được hợp nhất với kiểm tra khả năng tiếp cận. Ngoài ra còn có một số công cụ cần phải cài đặt trên phần cứng hoặc máy chủ. Những công cụ này hữu ích cho việc kiểm thử các ứng dụng web lớn và phức tạp.

**Phạm vi**

Phạm vi của một công cụ kiểm thử khả năng tiếp cận ứng dụng web được xác định bởi các tính năng của nó. Một số công cụ đánh giá, phân tích toàn bộ ứng dụng web và kiểm tra xem nó có tuân theo với các nguyên tắc về khả năng tiếp cận khác nhau không. Một số công cụ khác chỉ đánh giá khả năng truy cập của một trang duy nhất và đưa ra một báo cáo chi tiết về trang được kiểm tra. Ngoài ra còn có một số công cụ chỉ kiểm tra cụ thể một loại khả năng truy cập trong ứng dụng web. Ví dụ như kiểm tra tỷ lệ tương phản màu sắc, từ đó giải quyết các vấn đề liên quan đến loại khuyết tật cụ thể như mù màu, thị lực kém.

**Công cụ chỉ đánh giá và công cụ đánh giá và sửa lỗi**

Nhiều công cụ chỉ kiểm tra khả năng truy cập ứng dụng web mà không đưa ra có gợi ý để sửa lỗi, trong khi một số công cụ khác có thể đánh giá và sửa chữa các lỗi về khả năng truy cập từ đó giúp nhà phát triển rút ngắn thời gian để khắc phục các lỗi. Tính năng sửa chữa lỗi khả năng tiếp cận chỉ có trong các sản phẩm thương mại.

**So sánh các công cụ kiểm thử khả năng tiếp cận**

Khóa luận này chỉ tập chung nghiên cứu các công cụ kiểm thử khả năng tiếp cận miễn phí, độc lập và tuân theo tiêu chuẩn WCAG 2.0. Một số công cụ này được phát triển theo mã nguồn mở, từ đó giúp các nhà phát triển có thể chỉnh sửa, cập nhật và triển khai một cách linh hoạt. Cụ thể ba công cụ sẽ được khóa luận so sánh là Achecker, TAW và WAVE. Đây những những công cụ miễn phí, mã nguồn mở và được sử dụng rộng rãi hiện nay bởi các công ty phát triển phần mềm. Khóa luận tiến hành kiểm tra khả năng tiếp cận của các trang web khác nhau bằng cách sử dụng ba công cụ này. Các trang web được thử nghiệm có nội dung tiếng anh hoặc tiếng việt với các loại nội dung web khác nhau như là về tin tức, giải trí hay giáo dục. Cả ba công cụ này đều được sử dụng trên nền tảng web, người dùng chỉ cần truy cập vào mã định danh tài nguyên (URI) của mỗi công cụ để có thể sử dụng chúng. AChecker sau kiểm thử ứng dụng web tạo ra ba loại lỗi, cụ thể là lỗi xác định, lỗi có khả năng xảy ra và lỗi tiềm ẩn. Sau khi đánh giá, phân tích ứng dụng web WAVE đưa ra các loại lỗi, cảnh báo, tính năng, cấu trúc các phần tử của trang web và cả các lỗi tương phản. Các lỗi khả năng tiếp cận được phát hiện bởi TAW được phân loại thành lỗi xác định, cảnh báo và không được xem xét. Trong ba công cụ kiểm thử trên, công cụ duy nhất hiển thị các nguyên tắc đánh giá về khả năng tiếp cận được triển khai là AChecker trong khi TAW và WAVE ẩn các quy trình và nguyên tắc đánh giá khả năng truy cập web của chúng. Sau khi tiến hành kiểm thử khả năng tiếp cận của nhiều trang web khác nhau sử dụng Achecker, TAW và WAVE, kết quả về số lượng lỗi được phát hiện bởi các công cụ kiểm thử được biểu diễn trong bảng 3 bên dưới.

Bảng 3. Kết quả thử nghiệm các công cụ

|  |  |  |  |
| --- | --- | --- | --- |
| **Website** | **Achecker** | **Taw** | **Wave** |
| https://courses.uet.vnu.edu.vn/ | 178 | 90 | 16 |
| https://vnexpress.net/ | 425 | 546 | 68 |
| https://cafef.vn/ | 950 | 425 | 428 |
| https://www.w3schools.com/ | 557 | 387 | 63 |
| https://dantri.com.vn/ | 1574 | 40 | 261 |
| https://giaoduc.net.vn/ | 755 | 292 | 116 |
| https://hanoi.edu.vn/ | 642 | 280 | 106 |

Phân tích những kết quả này, ta có thể nhận xét rằng số lỗi được phát hiện ra bởi các công cụ kiểm thử là khác nhau đáng kể. Trong hầu hết các trang web, AChecker phát hiện ra số lượng lớn các vấn đề về khả năng truy cập hơn TAW và WAVE. TAW tạo ra các lỗi và hiển thị số lượng lỗi vi phạm theo nguyên tắc về khả năng tiếp cận. WAVE hiển thị snapshot của trang web và hiển thị lỗi vi phạm khả năng truy cập bằng các biểu tượng trên đó. Vì AChecker có hiệu suất tốt hơn trong việc phát hiện lỗi vi phạm khả năng tiếp cận ứng dụng web, khóa luận tập trung nghiên cứu về AChecker và các cách để cải thiện hiệu suất và tính tương tác của nó.

AChecker là một công cụ kiểm thử khả năng truy cập ứng dụng web được phát triển dưới dạng mã nguồn mở. Mục đích của công cụ này là phân tích khả năng truy cập của các ứng dụng web dựa trên nhiều loại của các nguyên tắc về khả năng tiếp cận của ứng dụng web quốc tế, chẳng hạn như Sec 508, BITV, STANCA, WCAG 1.0 và WCAG 2.0. Tuy nhiên khóa luận này chỉ tập trung vào nghiên cứu Achecker với tiêu chuẩn WCAG 2.0.

## 3.2 Phân tích công cụ kiểm thử AChecker

Các vấn đề của AChecker được phát hiện bằng cách tiến hành thử nghiệm kiểm tra các trang web khác nhau bằng công cụ AChecker và sử dụng cơ sở dữ liệu của nó để nghiên cứu các kĩ thuật đánh giá, phân tích lỗi ứng dụng web mà Achecker triển khai.

Các vấn đề của AChecker sau khi tiến hành thử nghiệm được phân loại thành ba nhóm. Nhóm thứ nhất là nhóm lỗi không rõ ràng, các lỗi được AChecker đưa ra một cách nhập nhằng, gây khó hiểu cho người dùng. Nhóm thứ hai là nhóm lỗi dư thừa, các lỗi Achecker phân tích được tồn tại nhiều lỗi trùng lặp với nhau. Nhóm thứ ba là nhóm lỗi thiếu sự gợi ý sửa lỗi, điều này khiến cho việc khắc phục lỗi trở lên khó khăn và tốn nhiều thời gian cho các nhà phát triển ứng dụng. Hơn nữa với những người không có kiến thức về khả năng tiếp cận ứng dụng web thì việc sửa lỗi gần như không thể thực hiện được. Sau khi tiến hành kiểm thử các trang web với Achecker, kết quả cho thấy các loại lỗi liên kết với phần tử web của nó như bảng 4. Phần lớn các vấn đề được phát hiện liên quan đến vấn đề sửa lỗi.

Bảng 4. Lỗi liên kết các phần tử web

|  |  |  |  |
| --- | --- | --- | --- |
| **Phần tử web** | **Lỗi dư thừa** | **Lỗi không rõ ràng** | **Thiếu sửa lỗi** |
| Image | ✓ | ✓ | ✓ |
| Link |  | ✓ |  |
| Frame |  |  | ✓ |
| Header |  | ✓ |  |
| Title |  |  | ✓ |
| Input | ✓ |  | ✓ |
| Table | ✓ |  | ✓ |
| Area code |  |  | ✓ |
| Html | ✓ |  | ✓ |
| Form | ✓ | ✓ |  |
| Anchor |  |  | ✓ |

Với thực trạng thực tế đã nêu ở trên, bài toán mà chúng ta cần giải quyết có 2 vấn đề:

1. Khắc phục lỗi không rõ ràng và lỗi trùng lặp xảy ra với công cụ Achecker.

2. Xây dựng thêm chức năng gợi ý sữa lỗi với giao diện trực quan. Qua đó mọi đối tượng người dùng có thể dễ dàng hiểu cách sửa lỗi cho từng trường hợp lỗi xảy ra mà không cần quá nhiều kiến thức chuyên môn, các nhà phát triển thì dễ dàng, đỡ tốn thời gian hơn trong quá trình khắc phục lỗi tiếp cận của ứng dụng web.

## 3.3 Xây dựng công cụ cải tiến AChecker

Để giải quyết vấn đề đã đặt ra, khóa luận xây dựng công cụ kiểm thử khả năng truy cập ứng dụng web MyChecker để cải thiện AChecker. Các lỗi được phát hiện bởi AChecker được phân loại theo các phần tử web có vấn đề về khả năng truy cập. Ví dụ các lỗi được phát hiện ra trên các thẻ như: image, link, frame, table, header, input, area, strong, label, keyboard, embedded, form. Khóa luận liệt kê một vài các vấn đề xảy ra với các phần tử web này và đề xuất giải pháp được triển khai trong MyChecker.

1. **Lỗi dư thừa**

Vấn đề về các lỗi được công cụ kiểm thử Achecker tạo ra sau khi kiểm thử các trang web là sự trùng lặp nhau, một vài lỗi xuất hiện lặp đi lặp lại hoặc những lỗi chứa nội dung vi phạm khả năng tiếp cận với ý nghĩa gần như nhau, điều này khiến cho người dùng cảm thấy rối và khó chịu trong khi đọc và phân tích lỗi. Cách tiếp cận của khóa luận để khắc phục vấn đề này là hợp nhất các lỗi trùng lặp thành một lỗi rõ ràng, loại bỏ đi những lỗi dư thừa, không cần thiết. Ví dụ với phần tử input, Achecker đưa ra hai lỗi với nội dung tương tự nhau là “phần tử input thiếu nhãn liên kết” và “nhãn của phần tử input không có nội dung”. Công cụ Mychecker gộp chúng lại thành một lỗi rõ ràng, dễ hiểu với người dùng. Bảng 5 dưới đây hiện thị một số lỗi trùng lặp và được khắc phục bởi công cụ MyChecker.

Bảng 5. MyChecker giải quyết các trường hợp lỗi trùng lặp

|  |  |  |
| --- | --- | --- |
| **Mã lỗi** | **AChecker** | **MyChecker** |
| 48 | Document language not identified | Language of the page is not specified |
| 49 | Document has invalid language code |
| 57 | Input element, type of "text", missing an associated label | The input element doesn't have associated label |
| 213 | Input element, type of "text", label has no text in label |
| 241 | Tabular information may be missing table markup | The information are not organized well |
| 248 | Visual lists may not be properly marked |
| 246 | All required form fields may not be indicated as required | Required fields are not specified |
| 267 | Form submission error messages may not identify empty required fields |
| 118 | Input element, type of "password", missing an associated label | The input element doesn't have associated label |
| 207 | Input element, type of "password", has no text in label |
| 120 | Input element, type of "file", missing an associated label | The input element doesn't have associated label |
| 216 | Input element, type of "file", has no text in label |
| 121 | Input element, type of "radio", missing an associated label | The input element doesn't have associated label |
| 204 | Input element, type of "radio", has no text in label |
| 119 | Input element, type of "checkbox", missing an associated label | The input element doesn't have associated label |
| 206 | Input element, type of "checkbox", has no text in label |
| 95 | Textarea element missing an associated label | The textarea element doesn't have associated label |
| 212 | Textarea has no text in label |

1. **Lỗi không rõ ràng**

Các lỗi không rõ ràng xảy ra khi mô tả lỗi không phù hợp hoặc khó hiểu. Cách tiếp cận MyChecker để cải thiện AChecker là thay thế những mô tả nhập nhàng, gây khó hiểu này bằng một mô tả đơn giản và rõ ràng mà bất kỳ người sử dụng nào cũng có thể hiểu được. Ví dụ với các lỗi với phần tử header, công cụ Achecker đưa ra thông báo về các lỗi này với nội dung là “phần tử header được sử dụng cho định dạng”, điều này khiến người đọc cảm thấy rất mơ hồ và khó hiểu. MyChecker sửa chúng thành “các phần tử header được sử dụng chia các phần trong nội dung trang web không phải được sử dụng để định dạng” giúp người dùng dễ hiểu hơn. Ngoài ra với nội dung lỗi “giá trị thuộc tính alt quá dài xảy ra trong phần tử img” thì người sử dụng cũng rất mơ hồ, không biết độ dài bao nhiêu là phù hợp. MyChecker thông báo lỗi với số lượng kí tự cụ thể cho giá trị thuộc tính “alt” điều này giúp cho người dùng biết cách chỉnh sửa lại nội dung sao cho phù hợp với độ dài yêu cầu. Bảng 6 dưới đây hiện thị một số lỗi không rõ ràng và được khắc phục bởi công cụ MyChecker.

Bảng 6. MyChecker giải quyết các trường hợp lỗi không rõ ràng

|  |  |  |
| --- | --- | --- |
| **Mã lỗi** | **AChecker** | **MyChecker** |
| 1 | Image Alt text may be too long | Limit of text in alt attribute is 100 characters |
| 42 | H1 may be used for formatting | Headers should be used for separating section not using for formatting |
| 43 | H2 may be used for formatting | Headers should be used for separating section not using for formatting |
| 44 | H3 may be used for formatting | Headers should be used for separating section not using for formatting |
| 45 | H4 may be used for formatting | Headers should be used for separating section not using for formatting |
| 46 | H5 may be used for formatting | Headers should be used for separating section not using for formatting |
| 47 | H6 may be used for formatting | Headers should be used for separating section not using for formatting |
| 268 | Form submission error messages may not provide assistance | Provide help for correcting the submitted errors |
| 38 | Header nesting - header following h2 is incorrect | H4 follow h2, headers should be used in order |

1. **Sửa lỗi**

Các lỗi được Achecker phân tích từ các ứng dụng web đưa ra là những đoạn văn bản cùng với những kiến thức liên quan về khả năng tiếp cận. Do đó để có thể khắc phục được lỗi, người dùng phải kinh nghiệm, kiến thức chuyên môn về kiểm thử khả năng truy cập ứng dụng web. Để giải quyết vấn đề này, công cụ MyChecker đề xuất chức năng sửa lỗi với giao diện và tính tương tác trực quan giúp các nhà phát triển thì dễ dàng, đỡ tốn thời gian hơn trong quá trình khắc phục lỗi tiếp cận của ứng dụng web. Ví dụ, lỗi thiếu thuộc tính “alt” xảy ra với phần tử img, MyChecker sẽ tự động thêm thuộc tính “alt” vào phần tử img đồng thời giá trị của thuộc tính này được nhập tùy chỉnh bởi người dùng. Khi ngôn ngữ không được chỉ định trong ứng dụng web, AChecker tạo ra hai lỗi trùng lặp với nội dung tương tự nhau: “ngôn ngữ nội dung trang web chưa được xác định”. Công cụ MyChecker khắc phục lỗi này bằng cách tự động thêm thuộc tính “lang” vào phần tử html và mã ngôn ngữ được nhập bởi người dùng đồng thời MyChecker cũng giải quyết vấn đề về lỗi dư thừa do AChecker tạo ra. Dựa trên tiêu chuẩn WCAG 2.0, phần tiêu đề là cần thiết cho mỗi trang web. AChecker tạo ra lỗi vi phạm khả năng tiếp cận trong hai trường hợp liên quan đến tiêu đề trang web. Trong trường hợp không có thẻ tiêu đề, MyChecker sẽ trực tiếp sửa lỗi bằng cách thêm thẻ tiêu đề với nội dung tiêu đề được nhập bởi người dùng vào trang web. Trong trường hợp nội dung tiêu đề để trống hoặc chứa nội dung không hợp lệ, công cụ MyChecker sẽ sửa lỗi bằng cách cho phép người dùng chỉnh sửa lại nội dung của thẻ tiêu đề. Ngoài ra, WCAG 2.0 đề cập rằng mỗi phần tử đầu vào phải có một nhãn liên kết để mô tả mục đích của đầu vào. AChecker tạo ra lỗi trong trường hợp thiếu nhãn liên kết. MyChecker đưa ra giải pháp bằng cách thêm phần tử nhãn và cho phép người dùng nhập nội dung nhãn liên kết để mô tả chi tiết về mục đích của đầu vào một cách rõ ràng hơn. Bảng 7 dưới đây hiện thị một số lỗi được khắc phục bởi công cụ MyChecker.

Bảng 7. Lỗi được sửa bởi MyChecker

|  |  |  |
| --- | --- | --- |
| Mã lỗi | Mô tả lỗi | MyChecker |
| 1 | Phần tử img thiếu thuộc tính “alt” | Tự động thêm thuộc tính “alt” và giá trị được nhập bởi người dùng |
| 7 | Phần tử img với thuộc tính “alt” không hợp lệ | Cho phép người dùng nhập lại giá trị thuộc tính “alt” |
| 58 | Phần tử img được dùng trong input thiếu thuộc tính “alt” | Tự động thêm thuộc tính “alt” và giá trị được nhập bởi người dùng |
| 165 | Phần tử embed thiếu thuộc tính “alt” | Tự động thêm thuộc tính “alt” và giá trị được nhập bởi người dùng |
| 166 | Phần tử embed với thuộc tính “alt” không hợp lệ | Cho phép người dùng nhập lại giá trị thuộc tính “alt” |
| 101 | Phần tử frame với thuộc tính “alt” không hợp lệ | Cho phép người dùng nhập lại giá trị thuộc tính “alt” |
| 213 | Phần tử input không có nhãn liên kết | Tự động thêm nhãn cho phần tử input và giá trị được nhập bởi người dùng |
| 91 | Phần tử select không có nhãn liên kết | Tự động thêm nhãn cho phần tử select và giá trị được nhập bởi người dùng |
| 208 | Phần tử select với nhãn không hợp lệ | Cho phép người dùng nhập lại giá trị nhãn của phần tử liên kết |
| 212 | Phần tử area không có nhãn liên kết | Tự động thêm nhãn cho phần tử area và giá trị được nhập bởi người dùng |
| 31 | Phần tử iframe không có thuộc tính “title” | Tự động thêm thuộc tính “title” và giá trị được nhập bởi người dùng |
| 201 | Phần tử iframe chứa thuộc tính “title” không hợp lệ | Cho phép người dùng nhập lại giá trị thuộc tính “title” |
| 190 | Phần tử anchor thiếu thuộc tính “title” | Tự động thêm thuộc tính “title” và giá trị được nhập bởi người dùng |
| 50 | Trang web thiếu tiêu đề | Tự động thêm phần tử title trong phần head và giá trị được nhập bởi người dùng |
| 51 | Trang web chứa tiêu đề không hợp lệ | Cho phép người dùng nhập lại giá trị phần tử title trong phần head |
| 48 | Ngôn ngữ nội dung trang web chưa được xác định | Tự động thêm thuộc tính “lang” vào thẻ html và giá trị được nhập bởi người dùng |
| 3 | Thuộc tính “alt” quá dài | Cho phép người dùng nhập lại giá trị thuộc tính “alt” và giới hạn độ dài là 100 kí tự |

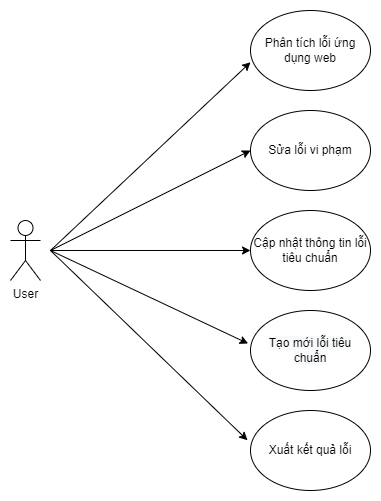
Sau khi phân tích các vấn đề của công cụ kiểm thử khả năng tiếp cận ứng dụng web AChecker và đưa ra giải pháp cho MyChecker, phần thiết kế hệ thống MyChecker được trình bày sau đây.

**Thiết kế hệ thống MyChecker**

Phần này trình bày chi tiết về các thành phần của công cụ kiểm thử khả năng tiếp cận được đề xuất trong khóa luận - MyChecker. Nó mô tả và giải thích các thành phần cần thiết trong hệ thống MyChecker. Ngoài ra các yêu cầu chức năng, phi chức năng của hệ thống và kiến trúc của nó cũng được miêu tả chi tiết và đầy đủ. Các yêu cầu chức năng được định nghĩa là các dịch vụ mà hệ thống phải thực hiện. Chúng được phân loại theo chức năng cụ thể của MyChecker. Các yêu cầu phi chức năng là các ràng buộc về an toàn, bảo mật, hiệu năng, mức độ khả dụng v.v mà hệ thống phải đáp ứng

**Yêu cầu chức năng**

Các chức năng chính của MyChecker là phân loại và sửa chữa các lỗi vi phạm khả năng tiếp cận ứng dụng web. Với mục đích này, một cơ sở dữ liệu mới được xây dựng chứa các loại lỗi với giải pháp sửa chúng. Các lỗi vi phạm khả năng tiếp cận đã được nghiên cứu và phân loại dựa trên các mô đun: mô đun lỗi không rõ ràng, mô đun lỗi dư thừa và mô đun sửa chữa lỗi để nâng cao kết quả khả năng tiếp cận ứng dụng web. Công cụ MyChecker được xây dựng với các dịch vụ, chức năng có thể tương tác trên giao diện web tạo điều kiện thuận lợi cho việc kiểm tra khả năng tiếp cận. Ngoài ra khi các lỗi vi phạm khả năng tiếp cận được giải quyết, kết quả thu được có thể được xuất ra tệp hoặc in giúp người dùng có thể quản lý và lưu dữ liệu lỗi vi phạm một cách tốt hơn. Hình 2 dưới đây biểu diễn các chức năng của công cụ được xây dựng.



Hình 2. Mô hình use-case của công cụ

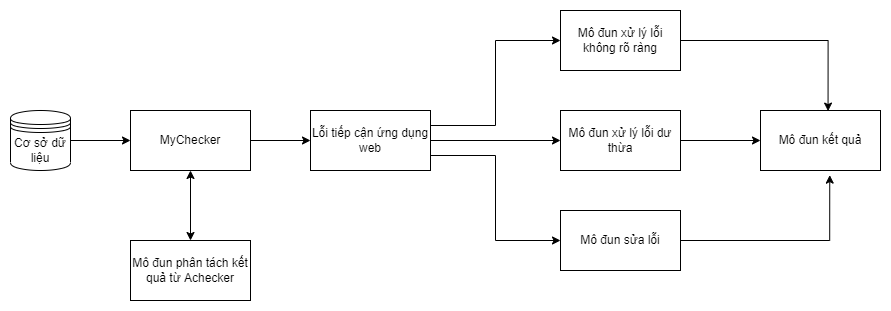
**Yêu cầu phi chức năng**

Các yêu cầu phi chức năng của MyChecker liên quan đến hiệu suất, hoạt động, độ tin cậy, khả năng bảo trì và khả năng tương tác, mức độ khả dụng. MyChecker được triển khai trên máy chủ node và cơ sở dữ liệu MySQL để truy xuất thông tin thuận tiện hơn. Ngoài ra, hệ thống phải cho phép nhiều người dùng có thể truy cập hệ thống đồng thời và người dùng không cần cài đặt bất kỳ phần mềm đặc biệt nào để sử dụng hệ thống. MyChecker hỗ trợ nhiều trình duyệt web khác nhau. Các yêu cầu phi chức năng khác được xem xét trong quá trình thiết kế và phát triển MyChecker được liệt kê bên dưới.

* Yêu cầu về hiệu năng: Thời gian phản hồi trung bình của hệ thống phải tương ứng với mức độ phức tạp của các đối tượng được kiểm tra.
* Yêu cầu về hoạt động: Ứng dụng có thể tạo ra các thông báo lỗi tự động trong trường hợp hệ thống trục trặc hay gặp sự cố bất ngờ.
* Yêu cầu về độ tin cậy: Các mô đun trong hệ thống phải cung cấp các thông báo chẩn đoán trong trường hợp hoạt động của các chức năng thành công, không thành công hoặc xảy ra lỗi.
* Yêu cầu về khả năng bảo trì: Tất cả các mô đun phần mềm được phát triển trong dự án sẽ được phát hành theo giấy phép mã nguồn mở.
* Yêu cầu về khả năng sử dụng: Khả năng sử dụng của hệ thống được tính toán theo hiệu quả và sự hài lòng về mặt định lượng và chất lượng.

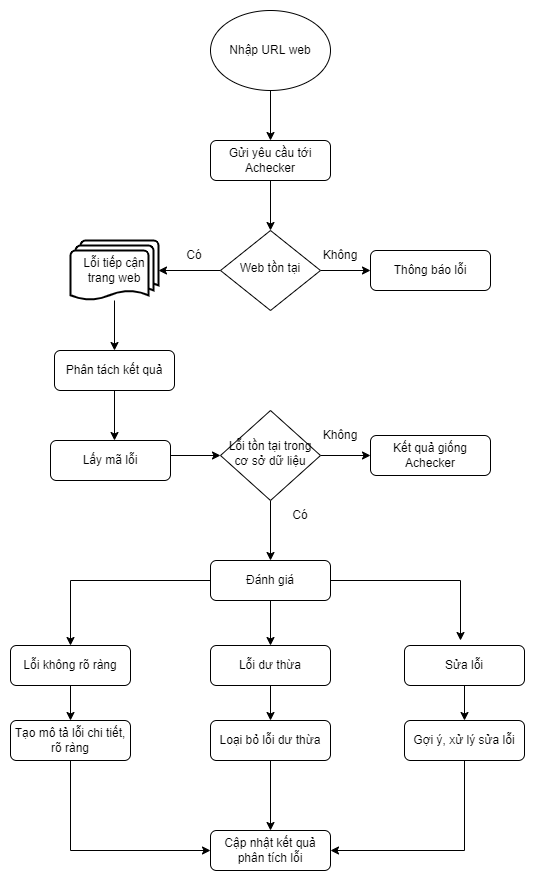
**Kiến trúc hệ thống**

Công cụ MyChecker là được xây dựng với năm mô đun chính bao gồm: mô đun phân tách kết quả từ AChecker, mô đun xử lý lỗi dư thừa, mô đun xử lý lỗi không rõ ràng, mô đun sửa lỗi và mô đun kết quả. Mỗi một mô đun trong ba mô đun đầu tiên có một vai trò cụ thể trong quá trình đánh giá, phân tích khả năng tiếp cận của ứng dụng web. Trong khi hai mô đun sửa chữa và mô đun kết quả liên quan đến mục tiêu sửa chữa lỗi vi phạm khả năng tiếp cận. Hình 3 bên dưới biểu diễn các thành phần trong kiến trúc của MyChecker.



Hình 3. Kiến trúc MyChecker

Trong mô đun phân tách kết quả, MyChecker giao tiếp với AChecker và nhận kết quả phân tích lỗi vi phạm khả năng tiếp cận ở định dạng HTML. MyChecker sử dụng thuật toán phân tích cú pháp từ các tệp kết quả lỗi HTML và lấy nội dung chi tiết của các lỗi từ các thẻ HTML rồi lưu chúng vào một mảng kết quả. Kết quả chứa ba thuộc tính là mã lỗi, mô tả lỗi và chi tiết lỗi. Mã lỗi xác định lỗi vi phạm khả năng truy cập bằng một số định danh duy nhất. Mô tả lỗi là mô tả chi tiết về lỗi vi phạm theo WCAG 2.0. Chi tiết lỗi là mã html của trang web có vấn đề về khả năng truy cập. Mô đun lỗi không rõ ràng xử lý các lỗi vi phạm khả năng tiếp cận là các lỗi có mô tả mơ hồ, không rõ ràng và thay thế mô tả của chúng thành các từ đơn giản, dễ hiểu và rõ ràng. Mô đun lỗi dư thừa kết hợp các lỗi có sự trùng lặp để chỉ tạo ra một lỗi duy nhất giúp tránh sự nhầm lẫn, rõ ràng trong quá trình đọc và phân tích lỗi. Mô đun sửa lỗi của MyChecker là một cách tiếp cận mới giúp sửa lỗi vi phạm khả năng tiếp cận ứng dụng web một cách hiệu quả hơn. Trong khi AChecker chỉ mô tả lỗi cũng như đề xuất cách khắc phục dựa trên văn bản, MyChecker sửa lỗi theo cách tương tác và cung cấp cho người dùng chức năng sửa lỗi một cách trực quan, dễ hiểu. Hơn nữa, quy trình sửa chữa được bán tự động để giúp các nhà phát triển dù không có nhiều kinh nghiệm về khả năng tiếp cận dễ dàng, đỡ tốn thời gian hơn trong quá trình khắc phục lỗi tiếp cận của ứng dụng web. Mô đun kết quả lưu trữ các lỗi vi phạm khả năng tiếp cận gặp phải, mỗi lỗi bao gồm các thông tin: mã định danh của lỗi vi phạm, mô tả lỗi và chi tiết về lỗi. Những kết quả phân tích lỗi vi phạm này được hiện thị trên một giao diện tương tác của MyChecker. Hơn nữa, kết quả có thể được xuất thành một báo cáo ở định dạng excel hỗ trợ người dùng quản lý và lưu trữ dữ liệu về kết quả phân tích lỗi tốt hơn. Hình 4 dưới đây là một sơ đồ cho thấy quy trình thực thi của năm mô đun trong MyChecker và sự tương tác giữa của nó với AChecker.



Hình 4. Flow chart của MyChecker

# Chương 4 Triển khai và đánh giá công cụ

Chương này trình bày về cách xây dựng công cụ kiểm thử khả năng tiếp cận ứng dụng web MyChecker để cải thiện công cụ Achecker hiện tại. Ngoài ra, trong chương này sẽ trình bày quá trình thực nghiệm, những đánh giá về công cụ cũng như là đưa ra những định hướng nghiên cứu, phát triển tiếp theo.

## Tiến hành xây dựng công cụ

Chúng ta tiến hành xây dựng công cụ hỗ trợ này dựa trên một ứng dụng Web, bao gồm Backend sử dụng RESTful API và Frontend sử dụng công nghệ SPA (Single-page application). Sau đây là chi tiết về mỗi phần.

### Backend

Backend được xây dựng dựa Web Framework Express - Framework đơn giản, phổ biến, sử dụng ngôn ngữ lập trình JavaScript đặc biệt là nó rất dễ sử dụng đối với các nhà phát triển. Backend gồm 3 phần chính: Phân tách kết quả từ Achecker; Xử lý lỗi không rõ ràng và lỗi dư thừa ; Sửa lỗi vi phạm khả năng tiếp cận. Sau đây là chi tiết về các thành phần này.

**Phân tách kết quả từ Achecker**

Công cụ MyChecker gửi thông tin URL trang web cần kiểm thử khả năng tiếp cận tới Achecker và nhận kết quả phân tích lỗi ở định dạng html. Ở đây, công cụ phân tích lỗi vi phạm khả năng tiếp cận MyChecker sẽ tiến hành phân tách kết quả phân tích lỗi được trả về từ công cụ Achecker. MyChecker tách các thuộc tính trả về từ kết quả phân tích lỗi, từ đó sử dụng các thuộc tính bao gồm: id lỗi, url trang web và đoạn mã html lỗi để phục vụ cho các mô đun phân tích, xử lý sau. Hình 5 là đoạn mã xử lý mô đun phân tách kết quả từ Achecker.



Hình 5. Phân tách kết quả phân tích lỗi

**Xử lý lỗi không rõ ràng và lỗi dư thừa**

Sau khi đã có có thông tin cần thiết của các lỗi vi phạm từ mô đun phân tách kết quả, công cụ sẽ tiến hành phân tích và xử lý các lỗi không rõ ràng, các lỗi dư thừa. Ở đây với mỗi mã lỗi sau khi được phân tách từ kết quả phân tích lỗi sẽ ứng với một lỗi tiêu chuẩn trong cơ sở dữ liệu MyChecker. Từ những nghiên cứu về những lỗi không rõ ràng, lỗi trùng lặp tồn tại với Achecker ở các chương trước, khóa luận tiến hành xây dựng một cơ sở dữ liệu mới cho công cụ MyChecker dựa vào cơ sở dữ liệu Achecker trước đó. Cơ sở dữ liệu này đã chỉnh sửa một số mô tả lỗi nhập nhằng, khó hiểu làm chúng trở lên rõ ràng hơn, hơn nữa với nhiều lỗi với nội dung tương tự nhau cơ sở dữ liệu đã lược bỏ những lỗi trùng lặp đó, gộp chúng lại thành một lỗi duy nhất. Từ đó cơ sở dữ liệu này đóng vai trò quan trọng trong mô đun xử lý lỗi dư thừa và lỗi không rõ ràng.

**Sửa lỗi vi phạm khả năng tiếp cận**

Ở phần trước, công cụ Mychecker đã xử lý lỗi dư thừa và lỗi không rõ ràng dựa vào mã lỗi từ kết quả phân tách lỗi. Mặc dù đã loại bỏ đi những sự nhập nhằng, không rõ ràng trong kết quả phân tích lỗi, tuy nhiên để người sử dụng có thể đọc và hiểu được nội dung lỗi vi phạm khả năng tiếp cận cũng như phương thức để sửa lỗi thì yêu cầu họ phải có kiến thức về khả năng tiếp cận ứng dụng web. Từ những bất cập này, chức năng sửa lỗi được xây dựng nhằm mục đích giúp người sử dụng có cái nhìn trực quan về phương pháp sửa lỗi vi phạm khả năng tiếp cận. Điều quan trọng là dù cho người dùng không có kiến thức chuyên môn về khả năng truy cập ứng dụng web vẫn có thể hiểu vào thao tác để sửa lỗi được. Mô đun sửa lỗi xử lý các trường hợp lỗi vi phạm tiêu chuẩn WCAG 2.0 dựa trên những gợi ý của từng lỗi vi phạm đó. Với từng loại lỗi sẽ được xếp vào mã sửa lỗi phù hợp, từ đó hệ thống có thể xử lý các lỗi vi phạm theo từng trường hợp đã được định nghĩa trong mô đun sửa lỗi. Ví dụ lỗi vi phạm với mã sửa lỗi có id bằng 1, đoạn mã xử lý sẽ thêm thuộc tính “alt” vào phần tử web vi phạm khả năng truy cập và giá trị của thuộc tính “alt” được gán với giá trị nhập vào của người dùng. Hình 6 dưới đây là đoạn mã xử lý các lỗi vi phạm dựa vào từng mã sửa lỗi tương ứng. Sau khi thực hiện sửa lỗi vi phạm người dùng sẽ nhận được đoạn mã html đã được sửa lỗi và đoạn mã trước khi sửa, dựa vào đó người dùng có thể so sánh chúng một cách trực quan để khắc phục lỗi nhanh chóng hơn. Ngoài ra công cụ còn đưa ra chi tiết về các thành phần của lỗi vi phạm bao gồm: tiêu chí vi phạm, phần tử web vi phạm và gợi ý sửa lỗi.

****

Hình 6. Sửa lỗi vi phạm khả năng tiếp cận

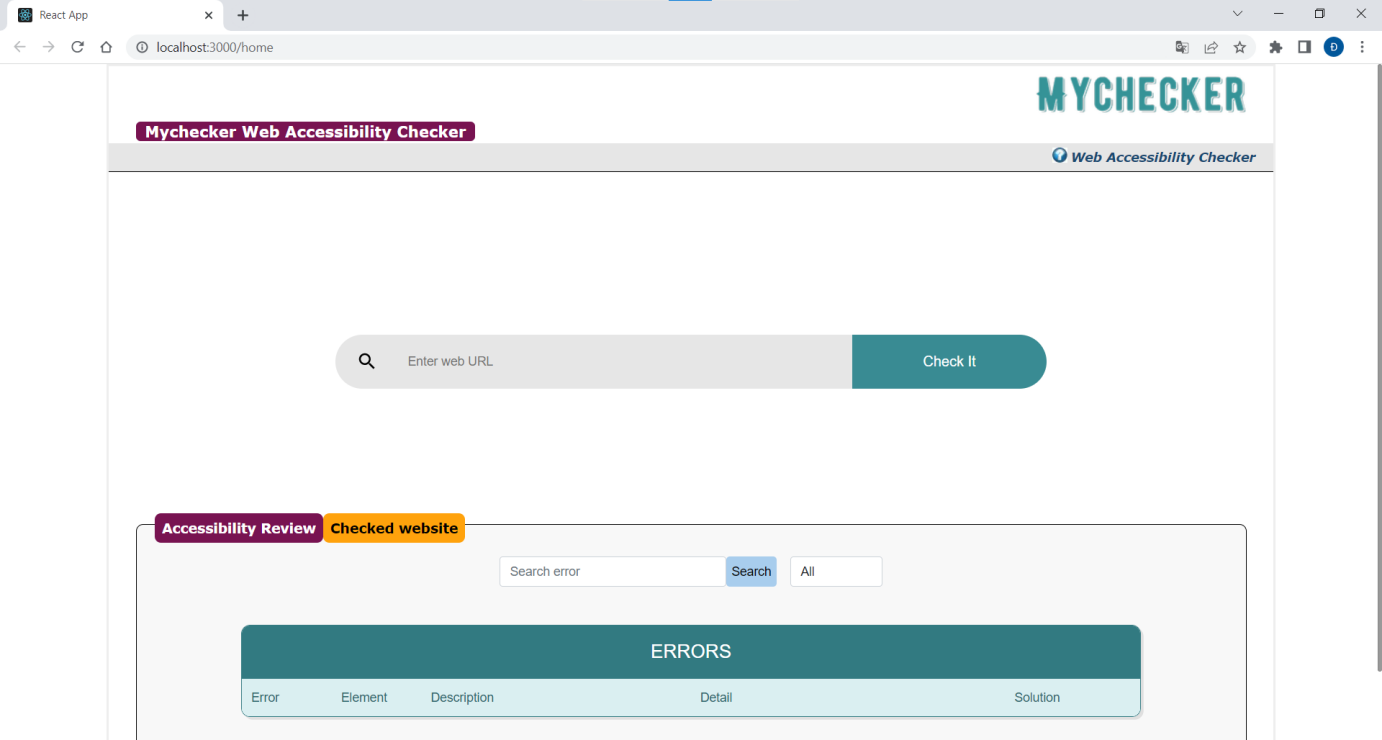
Để có thể đưa các chức năng của công cụ đến với người dùng, ta tiến hành đóng gói các APIs công khai. Bảng 8 là những APIs chính của Backend đã được xây dựng.

Bảng 8. APIs chính

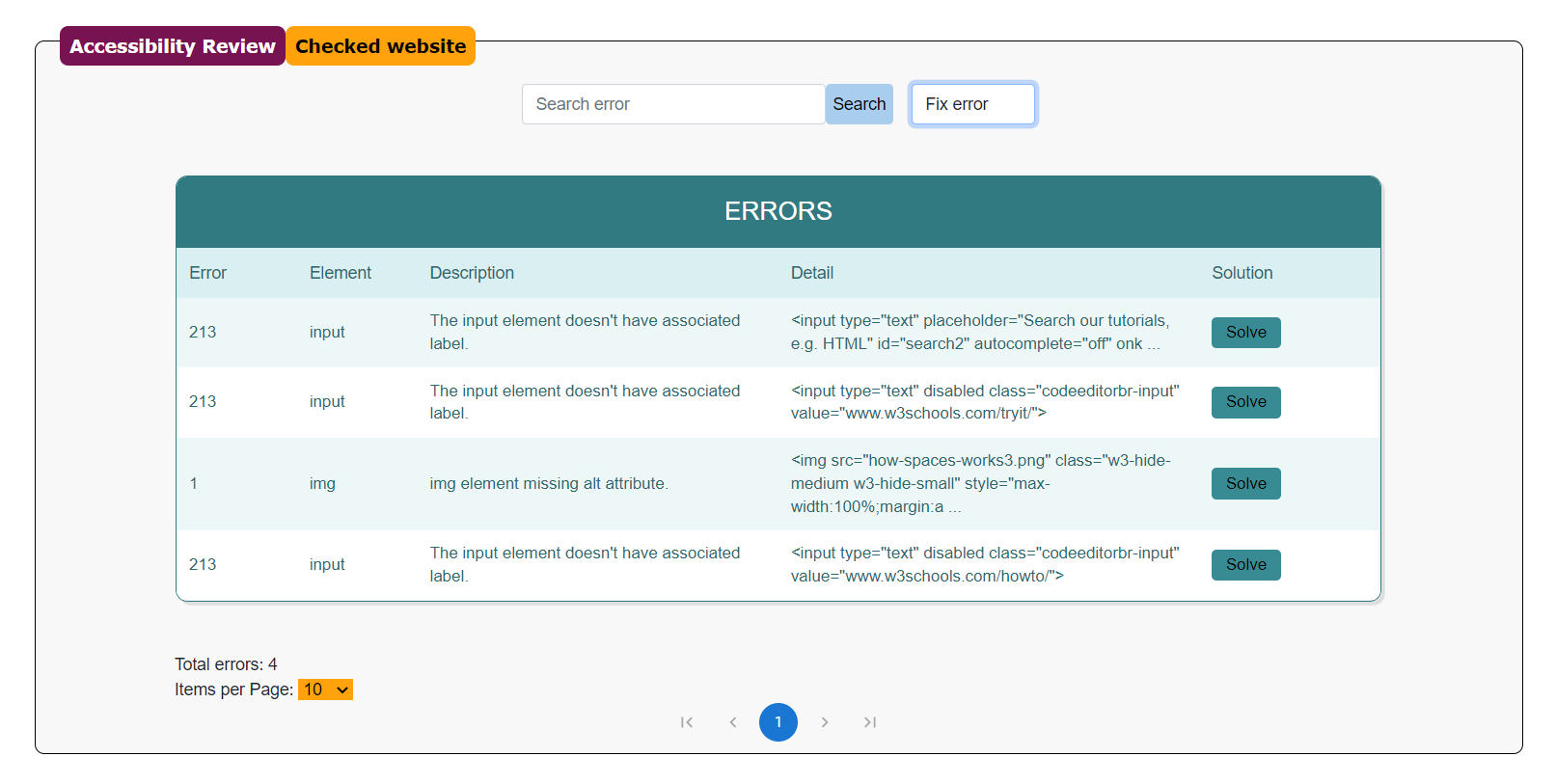
|  |  |  |
| --- | --- | --- |
| **API endpoint** | **Phương thức** | **Chức năng** |
| /api/check-web-url-error | POST | Tải thông tin URL web về backend và nhận kết quả phân tích lỗi |
| /api/fix-check-error | POST | Sửa lỗi vi phạm khả năng tiếp cận |
| /api/fetch-all-error | GET | Tải lên danh sách lỗi tiêu chuẩn |
| /api/update-error | PUT | Cập nhật thông tin lỗi |
| /api/create-new-error | POST | Tạo lỗi mới |

### Fontend

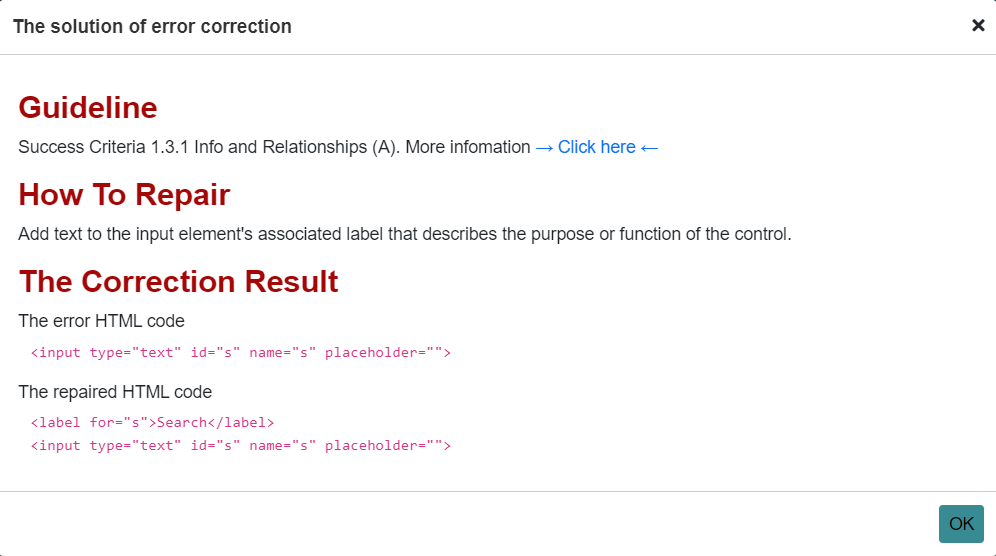
Ngày nay, khi ngành công nghệ càng ngày càng phát triển, số lượng các ứng công nghệ tăng nhanh chóng. Ta có thể thấy rất nhiều các ứng dụng khác nhau cùng biểu thị một nội dung hay cùng nhằm thỏa mãn một vài mục đích của chúng ta. Việc số lượng ứng dụng quá lớn như vậy dẫn đến tính cạnh tranh, lôi kéo người dùng giữa các ứng dụng này cũng rất lớn. Ngành Marketing Content có câu rất hay: “Chỉ có 6s để quyết định sự sống”. Nó khá chính xác với sự sống của đa số các ứng dụng hiện nay. Chỉ có 6s để thu hút một người dùng. Mà trong 6s ít ỏi đó thì cái đầu tiên người dùng nhìn thấy và trải nghiệm đó chính là giao diện của ứng dụng chứ không phải là toàn bộ nội dung bên trong nó. Vì vậy ta có thể thấy việc thiết kế giao diện người dùng quan trọng thế nào đến việc xây dựng bất kì một ứng dụng web nào. Do đó công cụ MyChecker có áp dụng những nguyên lý thiết kế cơ bản để có thể giúp hỗ trợ công cụ phân tích trở nên dễ dùng, dễ tương tác và thân thiện với người sử dụng hơn. Dưới đây là một số hình ảnh về giao diện của công cụ.



Hình 7. Giao diện chính



Hình 8. Giao diện kết quả phân tích lỗi



Hình 9. Giao diện kết quả sửa lỗi

Nhằm hỗ trợ công cụ phân tích trở nên dễ dùng, dễ tương tác thì giao diện đóng vai trò rất quan trọng. Frontend sẽ giúp Backend làm điều này - nó thu thập và truyền tải lại thông tin đến cho người dùng. Frontend sử dụng nền tảng React để xây dựng, bên cạnh đó sử dụng một số thư viện hỗ trợ cho việc phát triển nhanh hơn. Công cụ này có 3 ca sử dụng là phân tích lỗi khả năng tiếp cận của ứng dụng web, sửa lỗi tiếp cận ứng dụng web và quản lý thông tin lỗi tiếp cận ứng dụng web. Hình 10 thể hiện những ca sử dụng của công cụ MyChecker. Chi tiết về các ca sử dụng này như sau.

**Ca sử dụng phân tích lỗi khả năng tiếp cận của ứng dụng web**

Mô tả: Người dùng nhập vào form thông tin link ứng dụng web. Sau đó sẽ được trả về màn hình kết quả phân tích.

Actor tham gia: Người sử dụng công cụ.

Luồng sự kiện:

1. Người dùng nhập vào form thông tin : URL của ứng dụng web.
2. Người dùng nhận kết quả phân tích.

**Ca sử dụng sửa lỗi tiếp cận ứng dụng web**

Mô tả: Người dùng chọn lỗi tiếp cận rồi bấm vào nút sửa lỗi. Sau đó sẽ được trả về màn hình thông tin chi tiết về lỗi, gợi ý sửa lỗi và đoạn mã sau khi lỗi được sửa.

Actor tham gia: Người sử dụng công cụ.

Luồng sự kiện:

1. Bấm nút sửa lỗi.
2. Người dùng nhận kết quả chi tiết về lỗi và cách sửa lỗi.

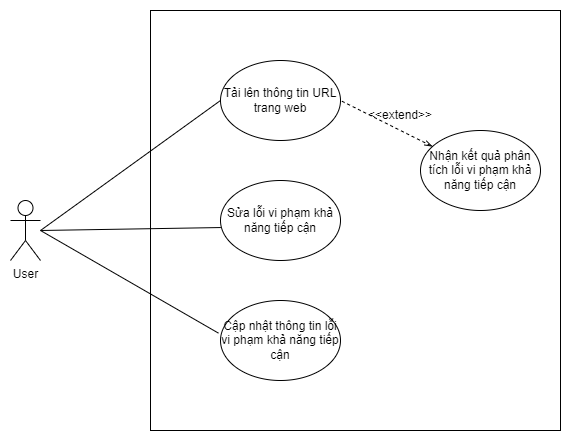
**Ca sử dụng quản lý thông tin lỗi tiếp cận ứng dụng web**

Mô tả: Người dùng chọn lỗi tiếp cận muốn cập nhật thông tin. Sau đó tiến hành sửa thông tin lỗi rồi bấm lưu để cập nhật thông tin lỗi.

Actor tham gia: Người phát triển công cụ.

Luồng sự kiện:

1. Chọn lỗi tiếp cận muốn cập nhật thông tin.
2. Cập nhật thông tin chi tiết về lỗi.
3. Bấm lưu để thay đổi thông tin chỉnh sửa.



Hình 10. Biểu đồ ca sử dụng của công cụ MyChecker

## Thử nghiệm công cụ

Trong phần này, khóa luận mô tả việc đánh giá công cụ được thực hiện trên MyChecker. Các chỉ số đo lường về mức độ hài lòng, sự hiệu quả và trải nghiệm người dùng được tính toán về mặt định lượng và chất lượng. Công cụ MyChecker tiến hành kiểm tra khả năng tiếp cận của nhiều trang web khác nhau. Các trang web được thử nghiệm có nội dung tiếng anh hoặc tiếng việt với đa dạng các loại nội dung web khác nhau.

Tiến hành thử nghiệm công cụ, chia thành hai giai đoạn thử nghiệm toàn bộ quy trình của công cụ và thử nghiệm mức độ cải thiện của công cụ. Cụ thể về quá trình thực nghiệm được trình bày sau đây.

**Thử nghiệm toàn bộ quy trình của công cụ**

Mục đích của việc thử nghiệm này là đánh giá về trải nghiệm người dùng. Thử nghiệm được tiến hành dựa trên quá trình người dùng sử dụng công cụ để kiểm thử khả năng tiếp cận của một ứng dụng web.

Quá trình thử nghiệm sử dụng công cụ để kiểm thử khả năng tiếp cận của một ứng dụng web trải qua các bước sau: (1) Nhập các thông tin cần thiết để kiểm thử khả năng tiếp cận của ứng dụng web. (2) Nhận kết quả phân tích trả về. (3) Sửa lỗi vi phạm. Người dùng dễ dàng sử dụng khi chỉ cần cung cấp đường dẫn URL trang web để phân tích lỗi khả năng tiếp cận. Sau khi quá trình phân tích được thực hiện xong kết quả phân tích lỗi được hiển thị dưới dạng bảng giúp người dùng dễ đọc hơn.

Mười nhà phát triển tham gia đã được chọn bao gồm nhân viên của các công ty phát triển phần mềm và sinh viên của các trường đại học. Tất cả đều không có kiến thức về khả năng tiếp cận, mặc dù họ có nhiều cấp độ kinh nghiệm lập trình khác nhau. Tất cả những người tham gia đã sử dụng cả AChecker và MyChecker làm công cụ đánh giá để kiểm tra khả năng truy cập của cùng một trang web. Các thử nghiệm đều được thực hiện trên cùng một máy tính chạy trên nền tảng hệ điều hành Windows 10 và trình duyệt web Google Chrome. Sau khi tiến hành các thử nghiệm, những người tham gia đã hoàn thành một cuộc khảo sát để đo mức độ hài lòng của họ dựa trên thang đo mức độ hài lòng. Kết quả về mức độ hài lòng của người dùng về MyChecker cho thấy công cụ mang lại sự hài lòng cao hơn so với AChecker. Các nhà phát triển nhiều kinh nghiệm có mức độ hài lòng cao nhất trung bình là 86%, 84% đối với các nhà phát triển chưa nhiều kinh nghiệm và 80% đối với các sinh viên đại học. Kết quả so sánh về mức độ hài lòng giữa công cụ Achecker và MyChecker được thể hiện trong hình 11.

Hình 11. Mức độ hài lòng thử nghiệm công cụ

**Thử nghiệm mức độ cải thiện của công cụ**

Quá trình thử nghiệm mức độ cải thiện công cụ được triển khai trên một số trang web khác nhau sử dụng công cụ MyChecker. Mục đích của việc thử nghiệm này là đưa ra những lỗi vi phạm được xử lý bởi MyChecker mà đã được cải tiến từ Achecker. Những lỗi vi phạm này nằm trong ba nhóm lỗi đã được phân tích trong khóa luận bao gồm: lỗi không rõ ràng, lỗi dư thừa, và sửa lỗi. Kết quả về số lượng lỗi trong các ứng dụng web được xử lý bởi công cụ MyChecker như bảng 9 dưới đây.

Bảng 9. Số lượng lỗi được xử lý bởi MyChecker

|  |  |  |  |
| --- | --- | --- | --- |
| **Website** | **Lỗi không rõ ràng** | **Lỗi dư thừa** | **Sửa lỗi** |
| https://courses.uet.vnu.edu.vn/ | 1 | 1 | 5 |
| https://vnexpress.net/ | 1 | 2 | 6 |
| https://cafef.vn/ | 1 | 2 | 3 |
| https://www.w3schools.com/ | 2 | 2 | 5 |
| https://uet.vnu.edu.vn/ | 2 | 1 | 12 |
| https://giaoduc.net.vn/ | 3 | 3 | 56 |
| https://hanoi.edu.vn/ | 2 | 1 | 49 |

## Đánh giá công cụ

Công cụ này vẫn đang trong giai đoạn phát triển, còn nhiều phần phải nghiên cứu, tối ưu và tìm hướng phát triển tốt hơn. Dưới góc nhìn của nhà phát triển công cụ, sau đây là một vài đánh giá về công cụ từ đó đưa ra hướng phát triển trong tương lai.

**Điểm nổi bật**

Với các yêu cầu đặt ra từ ban đầu, công cụ đã đáp ứng được cơ bản các yêu cầu đó. Công cụ được phát triển trên nền tảng ứng dụng Web hiện đại, có giao diện dễ tiếp cận và kết quả trả về dễ hiểu cho mọi đối tượng người dùng. Kết quả phân tích lỗi vi phạm khả năng tiếp cận được biểu diễn dưới dạng bảng với các thuộc tính lỗi được hiện thị một cách rõ ràng dễ đọc. Về chất lượng công cụ, theo thực nghiệm được trình bày phía trên thì MyChecker có những cải tiến rõ rệt so với việc sử dụng công cụ Achecker. Cụ thể là MyChecker đã giải quyết được những vấn đề về sai sót trong khi triển khai công cụ Achecker giúp loại bỏ những lỗi trùng lặp, lỗi không rõ ràng khi phân tích khả năng tiếp cận của ứng dụng web. Đặc biệt là chức năng sửa lỗi được phát triển ở công cụ MyChecker, chức năng này giúp người dùng có một cái nhìn trực quan, rõ ràng hơn về cách khắc phục lỗi vi phạm. Đối với riêng những nhà phát triển thì những gợi ý sửa lỗi một cách cụ thể, rõ ràng như vậy có thể giảm thiểu thời gian tìm kiếm và sửa lại vấn đề mà ứng dụng web gặp phải, từ đó họ có thể dành nhiều thời gian hơn để phát triển các tính năng khác của ứng dụng.

Ngoài ra, công cụ MyChecker sử dụng các công cụ, nền tảng và thư viện mở, nên các chi phí phát triển thấp - là điểm cộng để tiến hành áp dụng vào thực thế. Việc phân chia các phần rõ ràng, tường minh sẽ giúp cho việc phát triển thêm công cụ trong tương lai trở nên dễ dàng, đơn giản, mã nguồn của công cụ dễ đọc, dễ hiểu.

**Điểm tồn tại**

Công cụ vẫn đang trong quá trình nghiên cứu và phát triển, nên bên cạnh những điểm nổi bật đã đạt được, công cụ vẫn tồn tại những hạn chế cần được xử lý trong tương lai. Vấn đề đầu tiên của công cụ gặp phải là chức năng sửa lỗi ở công cụ mới chỉ xử lý được cái lỗi cơ bản trong hướng dẫn tiếp cận nội dung web (WCAG), còn tồn tại nhiều lỗi nâng cao cần phải cải tiến thêm công cụ để có thể xử lý được. Về mặt phạm vi, công cụ MyChecker mới chỉ kiểm thử khả năng tiếp cận của các ứng dụng web tuân theo tiêu chuẩn WCAG 2.0, còn các tiêu chuẩn như Stanca ACT và Section 508 vẫn chưa thể áp dụng để kiểm thử khả năng tiếp cận. Ngoài ra việc quản lý kết quả phân tích lỗi của từng ứng dụng web cũng là một vấn đề còn tồn tại ở công cụ MyChecker. Công cụ cần phải bổ sung thêm chức năng này để giúp người dùng có thể xem lại các kết quả phân tích một cách nhanh chóng hơn, tránh trường hợp phải thực hiện lại quá trình phân tích lỗi truy cập ứng dụng web từ đầu. Cuối cùng, giao diện công cụ hiện vẫn đơn giản, chưa có tính thẩm mỹ cao để thu hút người dùng.

Để có thể khắc phục được những điểm tồn tại của công cụ đã được trình bày ở phía trên, sau đây là những hướng phát triển trong tương lai. Mở rộng phạm vi kiểm thử khả năng tiếp cận của các ứng dụng web bằng cách áp dụng thêm các tiêu chuẩn như Stanca ACT và Section 508 vào quá trình phân tích sửa lỗi vi phạm khả năng truy cập. Chức năng quản lý kết quả phân tích lỗi cần phải được bổ sung sớm, từ đó công cụ có thể mở rộng thêm các tính năng về hiển thị kết quả phân tích lỗi dưới dạng các biểu đồ từ đó giúp người đọc có cái nhìn trực quan hơn về kết quả phân tích lỗi. Ngoài ra, cần tối ưu lại hệ thống để nâng cao hiệu năng của công cụ. Thiết kế lại giao diện người dùng cũng rất cần thiết. Công cụ có chức năng tuyệt vời nhưng khả năng tương tác với người dùng không tốt thì cũng sẽ không có tính cạnh tranh, khó giữ chân người dùng. Thành bại của một công cụ nằm phần lớn ở điểm này.

# Kết luận

Nghiên cứu của khóa luận này nhằm mục đích cải thiện các công cụ kiểm thử khả năng truy cập ứng dụng Web hiện có. Với mục đích này, khóa luận đã nghiên cứu các công cụ kiểm thử khả năng tiếp cận và nguyên tắc hỗ trợ truy cập web phổ biến hiện nay. Các công cụ được nghiên cứu là các công cụ miễn phí tuân theo nguyên tắc hỗ trợ khả năng truy cập nội dung Web WCAG 2.0. Khóa luận nghiên cứu phân tích công cụ kiểm thử AChecker, từ kết quả phân tích công cụ cho thấy những thiếu sót và bất cập còn tồn tại trong chính công cụ Achecker. Những vấn đề này được xác định là kết quả phân tích lỗi không rõ ràng, sự trùng lặp lỗi và thiếu chức năng gợi ý sửa lỗi.

Trong khóa luận này, một công cụ kiểm thử khả năng truy cập web mới, MyChecker, được thiết kế và triển khai dựa trên công cụ AChecker. MyChecker được tạo ra để khắc phục những thiếu sót được xác định trong AChecker. Cuối cùng, một nghiên cứu đánh giá khả năng sử dụng cho công cụ MyChecker đã được thực hiện, nghiên cứu này liên quan đến một nhóm các nhà phát triển web, có trình độ kiến thức khác nhau về lập trình web và không có kiến thức nền tảng về khả năng tiếp cận ứng dụng web. Mục đích của nghiên cứu này nhằm đánh giá khả năng sử dụng đối với người dùng dựa trên sự so sánh mức độ hài lòng và khả năng sử dụng hiệu quả của hai công cụ MyChecker và Achecker. Kết quả của nghiên cứu cho thấy mức độ hài lòng và khả năng sử dụng hiệu quả của MyChecker vượt trội hoàn toàn so với Achecker. Ngoài ra, một nghiên cứu thống kê về khả năng xử lý lỗi vi phạm khả năng truy cập web của MyChecker cũng được thực hiện để đánh giá mức độ hoàn thiện của công cụ thông qua khả năng giải quyết các vấn đề được xác định trong công cụ AChecker.

Trong phạm vi nghiên cứu của khóa luận, các giai đoạn là phân tích các công cụ hiện tại, thiết kế và phát triển một công cụ tương tác mới. Sau đó, công cụ được nhiều người dùng thử nghiệm và so sánh với AChecker. Các định hướng phát triển tương lai cho công cụ có thể có nhiều giai đoạn khác nhau. Khóa luận mới chỉ nghiên cứu các hướng dẫn của WCAG 2.0, các công việc trong tương lai có thể được mở rộng sang các hướng dẫn truy cập Web khác như BITV, Stanca ACT, và Mục 508. Hơn nữa, khóa luận có thể làm việc trên một công cụ sửa chữa hoàn toàn tự động, nơi các nhà phát triển sẽ không có quyền kiểm soát việc sửa chữa cho các lỗi vi phạm và việc sửa chữa các lỗi vi phạm diễn ra hoàn toàn tự động. Đây là một ý tưởng thú vị nhưng sẽ gặp những khó khăn nhất định trong khi thực hiện vì chỉ có một số lỗi vi phạm đủ đơn giản để có thể sửa chữa một cách tự động mà không cần đến sự can thiệp của người dùng.

**Tài liệu tham khảo**

1. Burgstahler, Jirikowic, Kolko and Elio, 2004. “Software Accessibility, Usability Testing and Individuals with Disabilities”, Information Technology and disabilities Journal.
2. Walter Kern, 2008. “Web 2.0 – End of accessibility? Analysis of Most Common Problems with Web 2.0 Based Applications Regarding Web Accessibility” International Journal of Public Information Systems.
3. Gagan Gurung, Rahul Shah, Dhiraj Prasad Jaiswal, 2020. "Software Development Life Cycle Models-A Comparative Study", International Journal of Scientific Research in Computer Science Engineering and Information Technology.
4. Meghan Mcconnell, Christina St-Onge, Meredith E Young, 2014. "The benefits of testing for learning on later performance", Advances in Health Sciences Education
5. Bambang Parmanto, Xiaoming Zeng, 2005. "Metric for Web accessibility evaluation", Journal of the American Society for Information Science and Technology
6. WHO. (2011). World report of disability. Retrieved from<http://www.who.int/disabilities/world_report/2011/report.pdf>
7. ISO. (2012). ISO 9241-171 Ergonomics of human-system interaction – Guidance on software accessibility. Retrieved from [https://www.iso.org/obp/ui/#iso:std:iso:9241:-171:ed-1:v1:en](https://www.iso.org/obp/ui/" \l "iso:std:iso:9241:-171:ed-1:v1:en)
8. WCAG 1.0. Available at <https://www.w3.org/TR/WAI-WEBCONTENT/>
9. Section 508. Available at <https://webaim.org/standards/508/checklist/>
10. WCAG 2.0. Available at <https://www.w3.org/TR/WCAG20/>
11. Owsley, R Sekuler, C Boldt, 2001. "Aging and low-contrast vision: face perception", Investigative ophthalmology & visual science.
12. WebAIM Million Report. Available at<https://webaim.org/blog/webaim-million-2021/>
13. WAVE- web accessibility evaluation. Available at <http://wave.webaim.org/>
14. TAW. Available at<https://www.tawdis.net/>
15. A Checker. Available at [http://www.achecker.ca](http://www.achecker.ca/)
16. Greg Gay, Cindy Qi Li, 2010. “AChecker: Open, Interactive, Customizable, Web Accessibility Checking, Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility Article