**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KĨ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----



**TIỂU LUẬN CUỐI KÌ**

**MÔN HỌC: KIỂM THỬ PHẦN MỀM**

**Đề tài:**

**XÂY DỰNG WEBSITE TUYỂN DỤNG**

**VÀ TÌM VIỆC LÀM**

**GVHD: ThS.** Nguyễn Trần Thi Văn

**Nhóm 09**

Phạm Ngọc Đăng Khoa 21110214

Nguyễn Hồng Thông Điệp 21110166

Trần Khải Hoàn 21110827

Đoàn Thái Sơn 21110289

Thành phố Hồ Chí Minh, tháng 12 năm 2024

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KĨ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----



**TIỂU LUẬN CUỐI KÌ**

**MÔN HỌC: KIỂM THỬ PHẦN MỀM**

**Đề tài:**

**XÂY DỰNG WEBSITE TUYỂN DỤNG**

**VÀ TÌM VIỆC LÀM**

**GVHD: ThS.** Nguyễn Trần Thi Văn

**Nhóm 09**

Phạm Ngọc Đăng Khoa 21110214

Nguyễn Hồng Thông Điệp 21110166

Trần Khải Hoàn 21110827

Đoàn Thái Sơn 21110289

Thành phố Hồ Chí Minh, tháng 12 năm 2024

**Nhận Xét Của Giảng Viên.**

*Hồ Chí Minh, ngày 15 tháng 12 năm 2024*

**Người Hướng Dẫn**

# LỜI CẢM ƠN.

Trước khi đi vào nội dung báo cáo, nhóm chúng em xin chân thành cảm ơn Trường Đại học Sư Phạm Kỹ Thuật Thành phố Hồ Chí Minh đã tạo cơ hội cho toàn thể sinh viên Khoa Công Nghệ Thông Tin nói chung và các thành viên trong nhóm nói riêng có cơ hội thực hiện tiểu luận chuyên ngành. Đây là cơ hội quý báu để nhóm em học hỏi và áp dụng kiến thức vào thực tế.

Chúng em cũng xin gửi lời cảm ơn chân thành đến Thầy Nguyễn Trần Thi Văn, người đã luôn tận tâm chỉ bảo và hỗ trợ nhóm chúng em trong suốt quá trình học môn Kiểm Thử Phần Mềm. Thầy đã truyền đạt cho nhóm không chỉ kiến thức chuyên môn mà còn là sự tự tin để thực hiện dự án lần này. Trong suốt 15 tuần học, thầy luôn đồng hành và hướng dẫn nhóm từ những bước đầu tiên đến khi hoàn thành tiểu luận, giúp nhóm đạt được những kết quả đáng khích lệ.

Tuy nhóm đã nỗ lực hết mình trong suốt quá trình học và thực hiện dự án, nhưng không tránh khỏi những thiếu sót về mặt kiến thức. Những thiếu sót đó có thể còn hiện hữu trong đồ án của nhóm. Nếu có bất kỳ sai sót nào, chúng em mong nhận được những phản hồi quý báu từ thầy cô và các bạn để nhóm có thể khắc phục và hoàn thiện hơn trong tương lai.

Cuối cùng, nhóm chúng em xin chúc thầy luôn mạnh khỏe, hạnh phúc và thành công trong công việc cũng như trong cuộc sống.

**Nhóm sinh viên thực hiện:**

Nguyễn Hồng Thông Điệp

Phạm Ngọc Đăng Khoa

Trần Khải Hoàn

Đoàn Thái Sơn

# MỤC LỤC

[PHẦN MỞ ĐẦU 1](#_Toc185191513)

[**1. Tính cấp thiết của đề tài 1**](#_Toc185191514)

[**2. Mục tiêu của đề tài 1**](#_Toc185191515)

[**3. Cách tiếp cận và nghiên cứu 2**](#_Toc185191516)

[**3.1. Đối tượng nghiên cứu 2**](#_Toc185191517)

[**3.2. Phạm vi nghiên cứu 4**](#_Toc185191518)

[**4. Ý nghĩa khoa học và thực tiễn 5**](#_Toc185191519)

[PHẦN NỘI DUNG 6](#_Toc185191520)

[**CHƯƠNG 1: PHÂN TÍCH YÊU CẦU DỰ ÁN 6**](#_Toc185191521)

[**1.1. Mô tả các yêu cầu đặt ra đối với dự án của nhóm. 6**](#_Toc185191522)

[**1.2. Phân tích yêu cầu 7**](#_Toc185191523)

[**1.3. Mô hình hóa yêu cầu 19**](#_Toc185191524)

[**CHƯƠNG 2: KẾ HOẠCH KIỂM THỬ DỰ ÁN. 67**](#_Toc185191525)

[**2.1. Tổng quan. 67**](#_Toc185191526)

[**2.2. Chiến lược. 70**](#_Toc185191527)

[**2.3. Tài nguyên. 72**](#_Toc185191528)

[**2.4. TEST MILESTONES. 73**](#_Toc185191529)

[**2.5. Tài liệu, báo cáo cần có. 74**](#_Toc185191530)

[**2.6. Đánh giá rủi ro. 74**](#_Toc185191531)

[**CHƯƠNG 3. CƠ SỞ LÝ THUYẾT VỀ KIỂM THỬ 75**](#_Toc185191532)

[**3.1. Các cấp độ kiểm thử 75**](#_Toc185191533)

[**3.2. Các phương pháp kiểm thử 90**](#_Toc185191534)

[**CHƯƠNG 4: KIỂM THỬ HỘP TRẮNG 102**](#_Toc185191535)

[**4.1. Hàm checkSeeCandiate. 102**](#_Toc185191536)

[**4.2. Hàm creatNewPackageCv. 107**](#_Toc185191537)

[**4.3. Hàm handleBanPost. 111**](#_Toc185191538)

[**4.4. Hàm handleSaveJobSkill. 117**](#_Toc185191539)

[**4.5. Hàm handleSaveJobLevel. 122**](#_Toc185191540)

[**4.6. Hàm handleChange. 126**](#_Toc185191541)

[**4.7. Hàm handleLogin (User). 130**](#_Toc185191542)

[**4.8. Hàm handleLogin (Company). 133**](#_Toc185191543)

[**CHƯƠNG 5: KIỂM THỬ HỘP ĐEN 137**](#_Toc185191544)

[**5.1. Tổng quan kết quả. 137**](#_Toc185191545)

[**5.2. Bug Report. 138**](#_Toc185191546)

[**CHƯƠNG 6: HỆ THỐNG QUẢN LÝ BUG 160**](#_Toc185191547)

[**6.1. Giới thiệu hệ thống Jira. 160**](#_Toc185191548)

[**6.2. Các tính năng nổi bật. 160**](#_Toc185191549)

[**6.3. Cách thức cài đặt và cấu hình. 162**](#_Toc185191550)

[**6.4. Ưu điểm và nhược điểm. 162**](#_Toc185191551)

[**6.5. Kết luận về Jira. 163**](#_Toc185191552)

[**6.6. Hình ảnh minh chứng Bug Report trên Jira. 163**](#_Toc185191553)

[PHẦN KẾT LUẬN 165](#_Toc185191554)

[**1. Những điều đã đạt được trong đề tài 165**](#_Toc185191555)

[**2. Ưu và nhược điểm của đề tài 166**](#_Toc185191556)

[**2.1. Ưu điểm 167**](#_Toc185191557)

[**2.2. Nhược điểm 167**](#_Toc185191558)

[**3. Hướng phát triển đề tài trong tương lai 167**](#_Toc185191559)

[TÀI LIỆU THAM KHẢO 169](#_Toc185191560)

[BẢNG PHÂN CHIA CÔNG VIỆC. 170](#_Toc185191561)

# KẾ HOẠCH THỰC HIỆN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên công việc** | **Người thực hiện** | **Ngày bắt đầu** | **Ngầy kế thúc** |
| 1 | Lựa chọn đề tài | Cả Nhóm | Oct 15, 2024 | Oct 18, 2024 |
| 2 | Đặc tả và mô hình hóa yêu cầu | Đoàn Thái Sơn,  Phạm Ngọc Đăng Khoa | Oct 19, 2024 | Nov 1, 2024 |
| 3 | Lập kế hoạch kiểm thử | Nguyễn Hồng Thông Điệp | Oct 19, 2024 | Oct 25, 2024 |
| 4 | Trình bày về các phương pháp kiểm thử đã áp dụng | Phạm Ngọc Đăng Khoa | Nov 2, 2024 | Nov 8, 2024 |
| 5 | Chọn mã nguồn để thực hiện kiểm thử hộp trắng | Cả Nhóm | Nov 9, 2024 | Nov 15, 2024 |
| 6 | Dựa vào 08 kỹ thuật kiểm thử hộp đen để thiết kế test case | Cả Nhóm | Nov 16, 2024 | Nov 29, 2024 |
| 7 | Lựa chọn và cài đặt một hệ thống quản lý lỗi | Trần Khải Hoàn | Nov 23, 2024 | Nov 29, 2024 |
| 8 | Hoàn Thành Báo Cáo | Nguyễn Hồng Thông Điệp. | Nov 30, 2024 | Dec 15, 2024 |

# PHẦN MỞ ĐẦU

1. **Tính cấp thiết của đề tài**

Kiểm thử phần mềm đóng vai trò quan trọng trong việc đảm bảo chất lượng của các hệ thống công nghệ thông tin hiện đại. Đặc biệt, trong bối cảnh các nền tảng trực tuyến ngày càng phát triển và được ứng dụng rộng rãi trong nhiều lĩnh vực, kiểm thử phần mềm không chỉ giúp phát hiện lỗi mà còn góp phần nâng cao hiệu suất và trải nghiệm người dùng.

Hệ thống tuyển dụng và tìm kiếm việc làm trực tuyến là một ví dụ điển hình về ứng dụng công nghệ thông tin trong thực tế. Việc xây dựng một nền tảng như vậy đòi hỏi các phương pháp kiểm thử chuyên sâu để đảm bảo hoạt động ổn định, bảo mật dữ liệu, và cung cấp trải nghiệm mượt mà cho người dùng. Bằng cách áp dụng các kỹ thuật kiểm thử hiện đại như kiểm thử chức năng, kiểm thử phi chức năng, và kiểm thử tự động hóa, chúng ta có thể đảm bảo rằng hệ thống đáp ứng đầy đủ yêu cầu của cả nhà tuyển dụng và ứng viên.

Trong bối cảnh nhu cầu kết nối giữa nhà tuyển dụng và người lao động ngày càng tăng, việc kiểm thử phần mềm để tối ưu hóa tính năng tự động sàng lọc CV là cần thiết. Điều này không chỉ cải thiện hiệu quả tuyển dụng mà còn giúp nâng cao độ tin cậy của hệ thống trong mắt người dùng.

1. **Mục tiêu của đề tài**

Mục tiêu của đề tài là xây dựng và kiểm thử một hệ thống tuyển dụng và tìm kiếm việc làm trực tuyến với các chức năng phục vụ công ty tuyển dụng, ứng viên và quản trị viên, đồng thời áp dụng các phương pháp kiểm thử phần mềm hiện đại để đảm bảo chất lượng toàn diện. Hệ thống cho phép công ty tuyển dụng đăng, chỉnh sửa, xóa và quản lý tin tuyển dụng, quản lý hồ sơ ứng viên và tự động sàng lọc CV dựa trên các tiêu chí như kỹ năng, trình độ học vấn, kinh nghiệm. Ứng viên có thể tìm kiếm việc làm theo ngành nghề, địa điểm, mức lương, nộp hồ sơ trực tuyến và nhận thông báo gợi ý việc làm phù hợp. Quản trị viên có quyền quản lý toàn bộ hệ thống, giám sát hoạt động của nhà tuyển dụng và ứng viên, xử lý vi phạm và bảo trì hệ thống.

Hệ thống sẽ được kiểm thử bằng các phương pháp kiểm thử hộp đen và hộp trắng. Kiểm thử hộp đen tập trung kiểm tra các chức năng chính của hệ thống dựa trên các test case được xây dựng từ yêu cầu phần mềm, như đăng ký, đăng nhập, tìm kiếm việc làm, nộp CV hay quản lý hồ sơ tuyển dụng. Kiểm thử hộp trắng sẽ tập trung vào kiểm tra logic bên trong mã nguồn, đảm bảo các hàm xử lý, vòng lặp và điều kiện hoạt động đúng. Các test case sẽ được xây dựng dựa trên phân lớp tương đương và phân tích giá trị biên, ví dụ kiểm tra đầu vào hợp lệ và không hợp lệ của chức năng đăng ký tài khoản hay tìm kiếm việc làm.

Quản lý lỗi sẽ được thực hiện bằng công cụ Jira để ghi nhận, theo dõi và xử lý lỗi trong quá trình kiểm thử. Quy trình quản lý lỗi bao gồm phát hiện, ghi báo cáo lỗi (bug report) với đầy đủ thông tin như tiêu đề, mức độ ưu tiên, mô tả lỗi, cách tái hiện lỗi và bằng chứng như ảnh chụp màn hình hoặc log file. Việc áp dụng các phương pháp kiểm thử và quản lý lỗi này không chỉ giúp phát hiện và khắc phục lỗi mà còn đảm bảo hệ thống hoạt động chính xác, ổn định, mang lại trải nghiệm tốt nhất cho người dùng và đáp ứng đúng các yêu cầu đặt ra.

1. **Cách tiếp cận và nghiên cứu**
   1. **Đối tượng nghiên cứu**

Để thực hiện kiểm thử với đề tài "Xây dựng website đăng tin tuyển dụng và tìm việc làm có tự động sàng lọc CV phù hợp," chúng em đã nghiên cứu các phương pháp kiểm thử phần mềm trong môi trường web, nhằm đảm bảo hệ thống hoạt động ổn định, hiệu quả và bảo mật. Mục tiêu của nghiên cứu này là xây dựng một bộ kiểm thử toàn diện cho website tuyển dụng, bao gồm các phương pháp kiểm thử hộp trắng, hộp đen, kiểm thử đơn vị, tích hợp, hệ thống và kiểm thử chấp nhận. Các yếu tố nghiên cứu chính của đề tài bao gồm:

* Kiểm thử hộp trắng (White-box testing): Phương pháp kiểm thử này tập trung vào việc kiểm tra cấu trúc bên trong của website tuyển dụng, kiểm tra các đoạn mã nguồn và đảm bảo tính chính xác trong việc xử lý các logic nghiệp vụ như đăng ký tài khoản, đăng nhập, và sàng lọc CV tự động.
* Kiểm thử hộp đen (Black-box testing): Mục đích là kiểm tra chức năng của hệ thống mà không cần biết đến mã nguồn. Kiểm thử hộp đen sẽ tập trung vào việc kiểm tra các chức năng chính của website như đăng tin tuyển dụng, nộp CV, và tìm kiếm việc làm, đảm bảo các tính năng này hoạt động đúng với yêu cầu đã đề ra.
* Kiểm thử đơn vị (Unit testing): Được thực hiện để kiểm tra các module riêng lẻ của website như chức năng đăng ký, đăng nhập, tìm kiếm việc làm, nộp CV, và các tính năng sàng lọc ứng viên. Mỗi chức năng sẽ được kiểm tra độc lập để đảm bảo tính ổn định và hiệu quả.
* Kiểm thử tích hợp (Integration testing): Mục đích của kiểm thử tích hợp là đảm bảo các phần của hệ thống hoạt động một cách liên kết và hiệu quả. Ví dụ, kiểm thử tích hợp sẽ kiểm tra tính tương thích giữa các module như quản lý hồ sơ ứng viên và sàng lọc CV tự động.
* Kiểm thử hệ thống (System testing): Kiểm tra toàn bộ hệ thống để đảm bảo tất cả các chức năng, tính năng và giao diện người dùng hoạt động chính xác và hiệu quả như yêu cầu của đề tài.
* Kiểm thử chấp nhận (Acceptance testing): Được thực hiện để đảm bảo website tuyển dụng đáp ứng được các yêu cầu và tiêu chuẩn do người dùng đặt ra, nhằm đảm bảo rằng hệ thống có thể được triển khai trong môi trường thực tế.
* Xây dựng test case: Việc xây dựng các test case cụ thể cho từng chức năng của website, ví dụ như kiểm tra việc đăng ký tài khoản, đăng nhập, sàng lọc CV, và tìm kiếm việc làm, đảm bảo các tính năng này hoạt động đúng và không có lỗi.
* Viết bug report và sử dụng công cụ quản lý bug: Khi phát hiện lỗi trong quá trình kiểm thử, bug report sẽ được ghi lại để mô tả chi tiết về lỗi và cách tái hiện nó. Công cụ quản lý bug sẽ được sử dụng để theo dõi tiến trình khắc phục lỗi và đảm bảo các vấn đề được giải quyết kịp thời.
  1. **Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài tập trung vào việc kiểm thử phần mềm cho website tuyển dụng và tìm việc làm trực tuyến, bao gồm các chức năng chính của hệ thống. Mục tiêu là đảm bảo tính ổn định, bảo mật và hiệu quả của hệ thống thông qua các phương pháp kiểm thử sau:

* + Kiểm thử chức năng người dùng: Bao gồm các chức năng như đăng ký tài khoản, đăng nhập, khôi phục mật khẩu, và quản lý hồ sơ người dùng. Mục đích là kiểm tra việc thực hiện các thao tác này có đúng như yêu cầu hay không.
  + Kiểm thử chức năng nhà tuyển dụng: Kiểm thử các chức năng như đăng tin tuyển dụng, tìm kiếm và phân loại tin tuyển dụng, và tự động sàng lọc ứng viên. Cần đảm bảo nhà tuyển dụng có thể sử dụng các công cụ này một cách chính xác và hiệu quả.
  + Kiểm thử hồ sơ ứng viên: Kiểm tra các chức năng cho phép ứng viên tạo, cập nhật và quản lý hồ sơ cá nhân, nộp CV trực tuyến. Các nhà tuyển dụng có thể truy cập và xem xét hồ sơ, từ đó thực hiện sàng lọc ứng viên phù hợp.
  + Kiểm thử tự động sàng lọc hồ sơ: Đảm bảo thuật toán sàng lọc ứng viên tự động hoạt động chính xác và đánh giá đúng các tiêu chí của nhà tuyển dụng, giúp tối ưu hóa quá trình tuyển dụng.
  + Kiểm thử chức năng tìm kiếm: Kiểm tra tính năng tìm kiếm việc làm của ứng viên và tìm kiếm hồ sơ ứng viên của nhà tuyển dụng, bao gồm các bộ lọc theo ngành nghề, vị trí, mức lương và kinh nghiệm.

Tóm lại, đề tài tập trung vào việc phát triển và thực hiện các kiểm thử phần mềm cho website tuyển dụng nhằm đảm bảo hệ thống đáp ứng các yêu cầu chức năng và hiệu suất, đồng thời giúp phát hiện và khắc phục các lỗi có thể xảy ra trong quá trình phát triển và sử dụng.

1. **Ý nghĩa khoa học và thực tiễn**

Sau khi hoàn thiện quá trình kiểm thử, chúng em sẽ tiếp tục rèn luyện và tiếp thu các đóng góp từ thầy cô và các chuyên gia để triển khai các chức năng nâng cao và phát triển hệ thống lên một bước phát triển mới. Mục tiêu là đưa website tuyển dụng vào ứng dụng thực tế, không chỉ phục vụ nhu cầu của môn học trong trường mà còn có thể mở rộng ra các đối tượng khác ngoài trường.

Việc kiểm thử sẽ giúp website tuyển dụng trở thành một nền tảng tin cậy và an toàn, giúp người dùng giao tiếp hiệu quả và nhanh chóng nhận thông báo. Các lỗi và sự cố sẽ được phát hiện và sửa chữa kịp thời, từ đó nâng cao chất lượng dịch vụ và cải thiện trải nghiệm người dùng.

Ngoài ra, việc phát triển và liên tục cải tiến hệ thống cũng sẽ góp phần vào việc nâng cao chất lượng công việc trong cộng đồng học thuật, tạo ra một nền tảng giao tiếp và kết nối mạnh mẽ giữa các sinh viên, giảng viên và nhà tuyển dụng.

# PHẦN NỘI DUNG

## **CHƯƠNG 1: PHÂN TÍCH YÊU CẦU DỰ ÁN**

### 1.1. Mô tả các yêu cầu đặt ra đối với dự án của nhóm.

Website tuyển dụng và tìm việc làm được tổ chức tổng quát như sau:

* Chức năng chung: Người dùng được phép đăng ký, đăng nhập, đăng xuất, quản lý thông tin cá nhân và tìm lại mật khẩu
* Một công ty có nhiều nhân viên bao gồm: nhóm nhà tuyển dụng và nhóm công ty
* Một nhân viên (nhóm công ty và nhóm nhà tuyển dụng) chỉ thuộc 1 công ty
* Nhóm công ty có thể mời nhóm nhà tuyển dụng chưa thuộc công ty vào công ty và có quyền thôi việc nhóm nhà tuyển dụng trực thuộc công ty
* Một bài viết thuộc về 1 công ty (xét theo nhân viên đăng bài) và 1 công ty có nhiều bài viết
* Một nhân viên trực thuộc công ty được phép quản lý thông tin bài đăng của công ty
* Một bài viết có 2 loại: Bình thường và nổi bật
* Một công ty các nhân viên có thể đăng nhiều bài viết phụ thuộc vào số lần còn lại của lượt đăng bài của từng loại (bình thường và nổi bật)
* Một gói bài đăng có số lượt đăng bài, giá tiền và loại bài viết được quản trị viên quy định
* Một công ty có thể mua lượt đăng bài theo từng loại (bình thường và nổi bật) và theo gói mà quản trị viên đã quy định
* Một gói xem ứng viên có số lượt xem, giá tiền
* Một công ty có thể mua lượt xem thông tin ứng viên tìm kiếm theo nhu cầu
* Quản trị viên có quyền thêm,sửa,xóa thông tin các danh mục
* Quản trị viên có quyền quản lý tài khoản người dùng
* Quản trị viên có quyền kiểm duyệt thông tin đăng ký của công ty
* Quản trị viên có quyền duyệt,từ chối,chặn bài viết của công ty
* Nhóm ứng viên được quyền tìm kiếm thông tin bài viết tuyển dụng mà quản trị viên đã duyệt.
* Nhóm ứng viên được quyền tìm kiếm thông tin các công ty đã đăng ký và được xem trạng thái kiểm duyệt công ty để có đánh giá về công ty đó
* Nhóm ứng viên được quyền nộp CV vào các bài viết tuyển dụng mà bản thân cảm thấy phù hợp
* Nhóm ứng viên được quyền xem lại danh sách CV đã nộp và xem chi tiết CV đã nộp
* Nhóm công ty và nhà tuyển dụng trực thuộc công ty được phép xem các CV đã nộp vào theo từng bài viết đã đăng của công ty
* Nhóm công ty và nhà tuyển dụng đã thuộc một công ty có thể tìm kiếm và xem thông tin ứng viên theo nhu cầu lựa chọn
* Nhóm ứng viên được quyền xem các thông tin công ty đã đăng ký
* Nhóm ứng viên được quyền xem các thông tin bài đăng đã được quản trị viên duyệt thành công
* Nhóm ứng viên có thể nộp CV ứng tuyển với bài viết tương ứng
* Nhóm ứng viên được phép cập nhật thông tin nâng cao của bản thân để chia sẻ thông tin đến nhà tuyển dụng khi họ có nhu cầu tìm kiếm và gợi ý công việc qua mail
* Nhóm ứng viên có thể lưu trữ CV online và nhà tuyển dụng có thể xem CV online đó khi nhóm ứng viên cho phép chia sẻ

### 1.2. Phân tích yêu cầu

#### 1.2.1. Yêu cầu chức năng nghiệp vụ

Yêu cầu chức năng nghiệp vụ (nhóm ứng viên)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Quy định/Công thức liên quan** | **Biểu mẫu liên quan** | **Ghi chú** |
| 1 | Đăng nhập | Tra cứu | - Tài khoản bắt buộc phải tồn tại trong database  - Bắt buộc phải điền số điện thoại và mật khẩu  - Nhận được quyền truy cập theo vai trò của tài khoản |  |  |
| 2 | Đăng ký | Lưu trữ | - Bắt buộc nhập đầy đủ thông tin email, họ, tên, số điện thoại, mật khẩu và xác nhận mật khẩu, giới tính  - Số điện thoại phải là duy nhất, chưa từng được đăng ký trước đây  - Email phải đúng định dạng, số điện thoại đúng 10 số  - Chọn hình thức đăng ký: ứng viên |  |  |
| 3 | Xem danh sách các công ty | Trích xuất | - Hiển thị tất cả các công ty đã được kích hoạt  - Có hỗ trợ tìm kiếm công ty theo tên |  |  |
| 4 | Xem danh sách các bài viết tuyển dụng | Trích xuất | - Hiển thị tất cả các bài viết tuyển dụng mà đã được kiểm duyệt  - Có hỗ trợ tìm kiếm theo tên bài viết và lọc thông tin theo nhu cầu (lĩnh vực, năm kinh nghiệm, lương,…) |  |  |
| 5 | Xem chi tiết thông tin công ty | Trích xuất | - Hiển thị thông tin chi tiết công ty đã được chọn  - Xem được dánh sách các bài tuyển dụng của công ty đã chọn |  |  |
| 6 | Xem chi tiết thông tin bài viết tuyển dụng | Trích xuất | - Hiển thị thông tin chi tiết bài viết tuyển dụng mà đã được chọn |  |  |
| 7 | Nộp CV | Lưu trữ | - Nộp CV của ứng viên theo bài viết tuyển dụng đã có chọn  - Có lời giới thiệu bản thân và file CV đính kèm |  |  |
| 8 | Xem danh sách CV đã nộp | Trích xuất | - Hiển thị tất cả các thông tin CV đã ứng tuyển |  |  |
| 9 | Xem chi tiết CV đã nộp | Trích xuất | - Có thể xem chi tiết thông tin về lời giới thiệu và file CV đính kèm đã gửi |  |  |
| 10 | Xem và thay đổi thông tin cá nhân | Trích xuất và lưu trữ | - Xem được thông tin đã đăng ký  - Có thể thay đổi thông tin đăng ký hoặc bổ sung thông tin đầy đủ thêm |  |  |
| 11 | Đổi mật khẩu | Lưu trữ | - Có thể đổi mật khẩu cũ sang mật khẩu mới |  |  |
| 12 | Quên mật khẩu | Lưu trữ | - Mã OTP sẽ gửi về số điện thoại đã đăng ký và tiến hành việc đổi mật khẩu mới |  |  |
| 13 | Xem và thay đổi thông tin nâng cao | Trích xuất và lưu trữ | - Điền các thông tin nâng cao và nộp CV online để chia sẻ đến nhà tuyển dụng và nhận gợi ý công việc qua mail |  |  |

Yêu cầu chức năng nghiệp vụ (nhóm nhà tuyển dụng)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Quy định/Công thức liên quan** | **Biểu mẫu liên quan** | **Ghi chú** |
| 1 | Đăng nhập | Tra cứu | - Tài khoản bắt buộc phải tồn tại trong database  - Bắt buộc phải điền số điện thoại và mật khẩu  - Nhận được quyền truy cập theo vai trò của tài khoản |  |  |
| 2 | Đăng ký | Lưu trữ | - Bắt buộc nhập đầy đủ thông tin email, họ, tên, số điện thoại, mật khẩu và xác nhận mật khẩu, giới tính  - Số điện thoại phải là duy nhất, chưa từng được đăng ký trước đây  - Email phải đúng định dạng, số điện thoại đúng 10 số  - Chọn hình thức đăng ký: nhà tuyển dụng |  |  |
| 3 | Đăng ký công ty | Lưu trữ | - Tài khoản nhà tuyển dụng phải chưa thuộc công ty  - Điền đầy đủ thông tin công ty như: Tên công ty, số lượng nhân viên, mô tả,…  - Hồ sơ chứng chỉ không bắt buộc điền nhưng nếu không điền sẽ mặc định là công ty ở trạng thái kiểm duyệt: Chưa kiểm duyệt.  - Nếu có điền hồ sơ thì trạng thái kiểm duyệt: Chờ kiểm duyệt  - Tạo thành công sẽ có 5 lượt đăng bài viết bình thường miễn phí  - Tạo thành công vai trò từ nhóm nhà tuyển dụng sẽ nâng cấp lên nhóm công ty |  |  |
| 4 | Đăng bài viết tuyển dụng | Lưu trữ | - Điền thông tin yêu cầu: lĩnh vực, yêu cầu kinh nghiệm, mức lương, địa điểm làm việc,…  - Có 2 lựa chọn loại bài viết: Nổi bật và bình thường. Khi đăng cần số lượt đăng bài của loại đã chọn tốn 1 lượt đăng nếu không có thì đăng thất bại  - Đăng bài thành công thì trạng thái kiểm duyệt: Chờ kiểm duyệt |  |  |
| 5 | Quản lý danh sách bài đăng | Trích xuất và lưu trữ | - Xem được danh sách các bài tuyển dụng của công ty bản thân trực thuộc  - Có thể thao tác: Sửa bài viết, xem ghi chú về việc thay đổi trạng thái kiểm duyệt bài viết  - Sửa các thông tin cho phép. Khi sửa bài viết sẽ quay về trạng thái kiểm duyệt: Chờ kiểm duyệt |  |  |
| 6 | Xem danh sách CV đã nộp | Trích xuất | - Hiển thị tất cả các thông tin CV đã ứng tuyển theo từng bài viết |  |  |
| 7 | Xem chi tiết CV đã nộp | Trích xuất | - Có thể xem chi tiết thông tin về lời giới thiệu và file CV đính kèm đã gửi |  |  |
| 8 | Tìm kiếm ứng viên theo yêu cầu | Trích xuất | - Có thể xem chi tiết thông tin và CV của ứng viên đã chia sẻ mà phù hợp với yêu cầu của mình |  |  |
| 9 | Xem và thay đổi thông tin cá nhân | Trích xuất và lưu trữ | - Xem được thông tin đã đăng ký  - Có thể thay đổi thông tin đăng ký hoặc bổ sung thông tin đầy đủ thêm |  |  |
| 10 | Đổi mật khẩu | Lưu trữ | - Có thể đổi mật khẩu cũ sang mật khẩu mới |  |  |
| 11 | Quên mật khẩu | Lưu trữ | - Mã OTP sẽ gửi về số điện thoại đã đăng ký và tiến hành việc đổi mật khẩu mới |  |  |
| 12 | Xem thống kê | Trích xuất | - Xem thống kê những lĩnh vực bài đăng nhiều nhất  - Xem thống kê số lượng CV được nhận theo từng bài viết theo khoảng thời gian |  |  |

Yêu cầu chức năng nghiệp vụ (nhóm công ty)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Quy định/Công thức liên quan** | **Biểu mẫu liên quan** | **Ghi chú** |
| 1 | Đăng nhập | Tra cứu | - Tài khoản bắt buộc phải tồn tại trong database  - Bắt buộc phải điền số điện thoại và mật khẩu  - Nhận được quyền truy cập theo vai trò của tài khoản |  |  |
| 2 | Đăng ký tài khoản cho nhóm nhà tuyển dụng và nhóm công ty trực thuộc công ty | Lưu trữ | - Bắt buộc nhập đầy đủ thông tin email, họ, tên, số điện thoại, giới tính, địa chỉ,…  - Số điện thoại phải là duy nhất, chưa từng được đăng ký trước đây  - Email phải đúng định dạng, số điện thoại đúng 10 số  - Chọn hình thức đăng ký: nhà tuyển dụng và công ty  - Mật khẩu sẽ được tự sinh ra và gửi trực tiếp đến mail đã đăng ký |  |  |
| 3 | Cập nhật công ty | Lưu trữ | - Cập nhật thông tin công ty như: Tên công ty, số lượng nhân viên, mô tả,…  - Nếu lúc tạo chưa có hồ sơ chứng chỉ nhưng khi cập nhật có bổ sung hồ sơ chứng chỉ thì trạng thái kiểm duyệt: Chờ kiểm duyệt. Nếu vẫn không bổ sung thì trạng thái kiểm duyệt: Chưa kiểm duyệt  - Thay đổi bất kì thông tin nào thì chuyển về trạng thái: Chờ kiểm duyệt nếu trước đó có trạng thái kiểm duyệt: Đã kiểm duyệt |  |  |
| 4 | Đăng bài viết tuyển dụng | Lưu trữ | - Điền thông tin yêu cầu: lĩnh vực, yêu cầu kinh nghiệm, mức lương, địa điểm làm việc,…  - Có 2 lựa chọn loại bài viết: Nổi bật và bình thường. Khi đăng cần số lượt đăng bài của loại đã chọn tốn 1 lượt đăng nếu không có thì đăng thất bại  - Đăng bài thành công thì trạng thái kiểm duyệt: Chờ kiểm duyệt |  |  |
| 5 | Quản lý danh sách bài đăng | Trích xuất | - Xem được danh sách các bài tuyển dụng của công ty bản thân trực thuộc  - Có thể thao tác: Sửa bài viết, xem ghi chú về việc thay đổi trạng thái kiểm duyệt bài viết  - Sửa các thông tin cho phép. Khi sửa bài viết sẽ quay về trạng thái kiểm duyệt: Chờ kiểm duyệt |  |  |
| 6 | Xem danh sách CV đã nộp | Trích xuất | - Hiển thị tất cả các thông tin CV đã ứng tuyển theo từng bài viết |  |  |
| 7 | Xem chi tiết CV đã nộp | Trích xuất | - Có thể xem chi tiết thông tin về lời giới thiệu và file CV đính kèm đã gửi |  |  |
| 8 | Tìm kiếm ứng viên theo yêu cầu | Trích xuất | - Có thể xem chi tiết thông tin và CV của ứng viên đã chia sẻ mà phù hợp với yêu cầu của mình |  |  |
| 9 | Xem và thay đổi thông tin cá nhân | Trích xuất và lưu trữ | - Xem được thông tin đã đăng ký  - Có thể thay đổi thông tin đăng ký hoặc bổ sung thông tin đầy đủ thêm |  |  |
| 10 | Đổi mật khẩu | Lưu trữ | - Có thể đổi mật khẩu cũ sang mật khẩu mới |  |  |
| 11 | Quên mật khẩu | Lưu trữ | - Mã OTP sẽ gửi về số điện thoại đã đăng ký và tiến hành việc đổi mật khẩu mới |  |  |
| 12 | Xem thống kê | Trích xuất | - Xem thống kê những lĩnh vực bài đăng nhiều nhất  - Xem thống kê số lượng CV được nhận theo từng bài viết theo khoảng thời gian |  |  |
| 13 | Mua lượt đăng bài | Lưu trữ | - Lựa chọn gói bài đăng theo loại nổi bật hay bình thường  - Chọn gói bài đăng  - Chọn số lượng  - Thanh toán online qua Sand box Paypal |  |  |

Yêu cầu chức năng nghiệp vụ (nhóm quản trị viên)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Công việc** | **Loại công việc** | **Quy định/Công thức liên quan** | **Biểu mẫu liên quan** | **Ghi chú** |
| 1 | Đăng nhập | Tra cứu | - Tài khoản bắt buộc phải tồn tại trong database  - Bắt buộc phải điền số điện thoại và mật khẩu  - Nhận được quyền truy cập theo vai trò của tài khoản |  |  |
| 2 | Đăng ký tài khoản cho nhóm nhà tuyển dụng, nhóm ứng viên, nhóm quản trị viên | Lưu trữ | - Bắt buộc nhập đầy đủ thông tin email, họ, tên, số điện thoại, giới tính, địa chỉ,…  - Số điện thoại phải là duy nhất, chưa từng được đăng ký trước đây  - Email phải đúng định dạng, số điện thoại đúng 10 số  - Chọn hình thức đăng ký: nhà tuyển dụng, ứng viên, quản trị viên  - Mật khẩu sẽ được tự sinh ra và gửi trực tiếp đến mail đã đăng ký |  |  |
| 3 | Quản lý tài khoản | Trích xuất và lưu trữ | - Xem thông tin đăng ký của tài khoản  - Xem xét và quản lý trạng thái tài khoản hoạt động: kích hoạt hoặc không kích hoạt |  |  |
| 3 | Quản lý các công ty | Trích xuất và lưu trữ | - Xem thông tin đăng ký của công ty  - Xem xét và quản lý trạng thái công ty hoạt động: kích hoạt hoặc không kích hoạt  - Xem xét và quản lý trạng thái kiểm duyệt của công ty: Chưa kiểm duyệt, chờ kiểm duyệt, đã kiểm duyệt |  |  |
| 4 | Quản lý danh sách các bài viết | Trích xuất và lưu trữ | - Xem thông tin đăng ký của bài viết  - Xem xét và quản lý trạng thái kiểm duyệt: Chờ kiểm duyệt, đã bị từ chối, đã kiểm duyệt,đã bị chặn  - Ghi chú thông tin mỗi lần thao tác gửi thông báo đến mail chủ công ty (người đã đăng ký công ty) |  |  |
| 5 | Quản lý các danh mục | Trích xuất và lưu trữ | - Tạo các thông tin danh mục: Loại công việc, cấp bậc, kinh nghiệm, khoảng lương, hình thức làm việc  - Có thể chỉnh sửa thông tin danh mục  - Xem thông tin chi tiết các danh mục |  |  |
| 6 | Quản lý các gói bài đăng và gói xem ứng viên | Trích xuất và lưu trữ | - Có thể thêm và chỉnh sửa các gói bài đăng theo 2 loại nổi bật và bình thường  - Có thể mở trạng thái kinh doanh và ngừng kinh doanh các gói  - Xem thông tin các gói |  |  |
| 7 | Xem và thay đổi thông tin cá nhân | Trích xuất và lưu trữ | - Xem được thông tin đã đăng ký  - Có thể thay đổi thông tin đăng ký hoặc bổ sung thông tin đầy đủ thêm |  |  |
| 8 | Đổi mật khẩu | Lưu trữ | - Có thể đổi mật khẩu cũ sang mật khẩu mới |  |  |
| 9 | Quên mật khẩu | Lưu trữ | - Mã OTP sẽ gửi về số điện thoại đã đăng ký và tiến hành việc đổi mật khẩu mới |  |  |
| 10 | Xem thống kê | Trích xuất | - Xem thống kê những lĩnh vực bài đăng nhiều nhất  - Xem thống kê doanh thu các gói bài đăng theo khoảng thời gian |  |  |

#### 1.2.2. Yêu cầu chức năng hệ thống

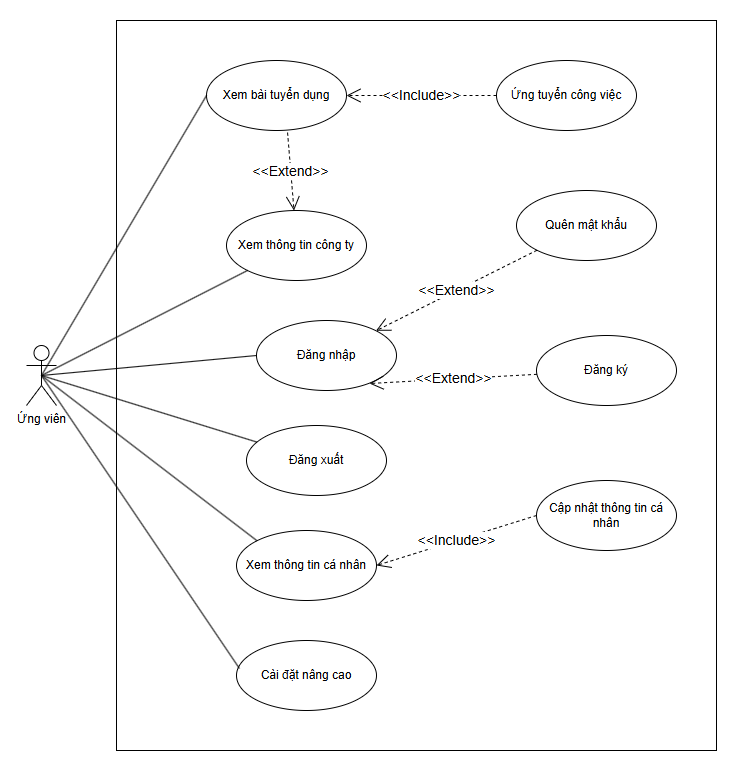
|  |  |  |
| --- | --- | --- |
| **Nội dung** | **Mô tả chi tiết** | **Ghi chú** |
| Phân quyền sử dụng | - Nhóm quản trị viên: xem tất cả công ty, bài đăng tin tuyển dụng; phê duyệt công ty, bài đăng tin; xử lý báo cáo vi phạm, quản lý thông tin tài khoản người dùng; quản lý các danh mục, gói bài đăng  - Nhóm công ty: Quản lý thông tin công ty, nhân viên thuộc công ty, mua gói bài đăng, đăng bài, xem danh sách CV và chi tiết CV, xem thống kê liên quan  - Nhóm tuyển dụng: Có thể tạo công ty nếu chưa trực thuộc công ty nào, được quyền đang bài khi đã thuộc công ty, xem danh sách CV và chi tiết CV, xem các thống kê liên quan  - Nhóm ứng viên: Tìm kiếm công ty, bài viết theo mong muốn, nhu cầu, ứng tuyển bằng cách nộp CV theo bài viết đã chọn, xem lại danh sách CV đã ứng tuyển và chi tiết CV đã nộp |  |

#### 1.2.3. Yêu cầu chất lượng

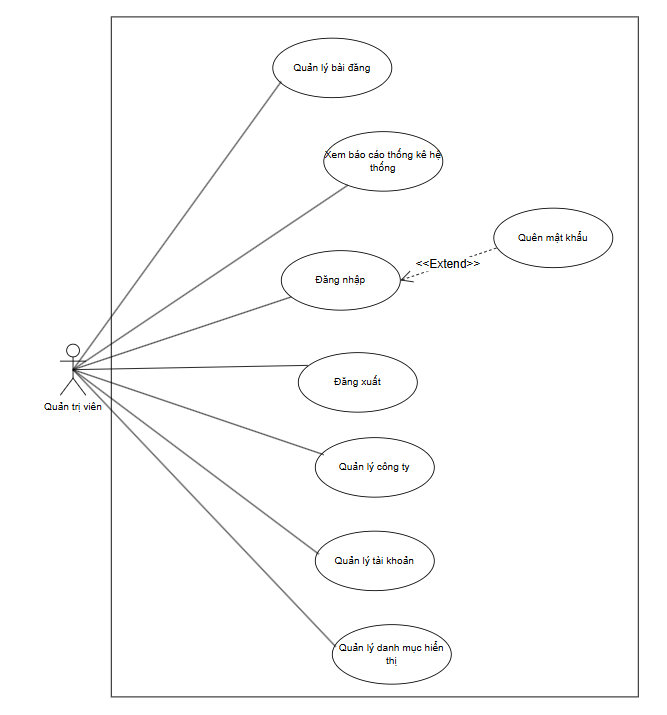
|  |  |  |
| --- | --- | --- |
| **Nội dung** | **Tiêu chuẩn** | **Mô tả chi tiết** |
| Sử dụng được trên tất cả các trình duyệt web | Tương thích | Chạy ổn trên các trình duyệt web hiện tại |
| Dễ dàng sử dụng, giao diện dễ nhìn thân thiện với người dùng | Tiện dụng | Người dùng có thể dễ dàng thao tác sử dụng mà không gặp bất kỳ khó khăn nào |
| Thông tin phải chính xác về nội dung và vị trí | Chính xác | Thông tin cần đảm bảo sự chính xác tuyệt đối, hình ảnh là ảnh thật |
| Thiết kế nhất quán | Thống nhất | Giao diện người dùng cần có thiết kế nhất quán ở mọi trang, bao gồm các thành phần như nút bấm, màu sắc, font chữ, để đảm bảo người dùng không bị bối rối khi chuyển giữa các trang khác nhau. |
| Tính ổn định | Ổn định | Hệ thống phải hoạt động ổn định trong suốt quá trình vận hành, không bị treo, dừng đột ngột hoặc gặp sự cố kỹ thuật gây gián đoạn trải nghiệm người dùng. |

### 1.3. Mô hình hóa yêu cầu

#### 1.3.1. Lược đồ Usecase



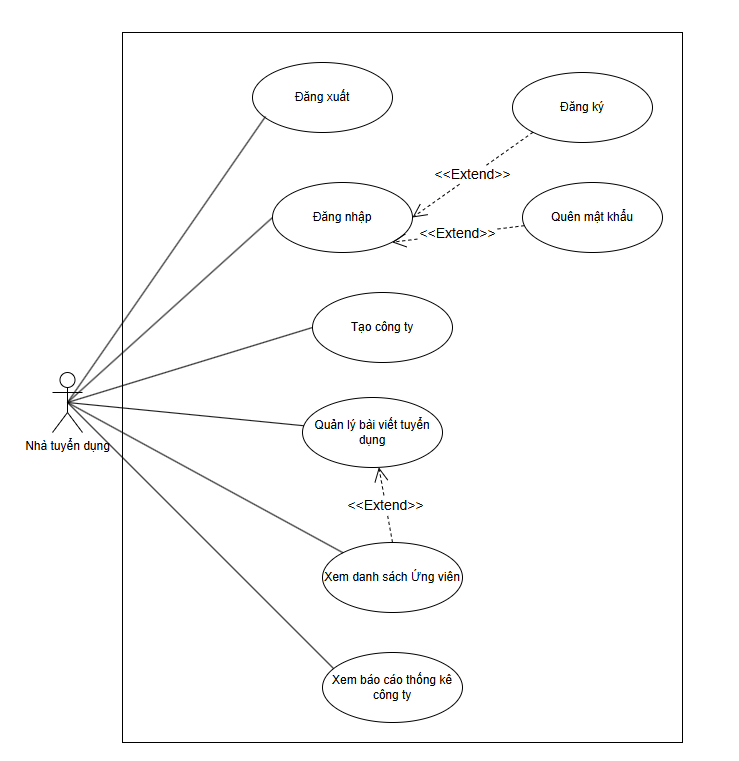
**Lược đồ Usecase nhóm Ứng viên**



**Lược đồ Usecase nhóm Quản trị viên**



**Lược đồ Usecase nhóm Công ty**



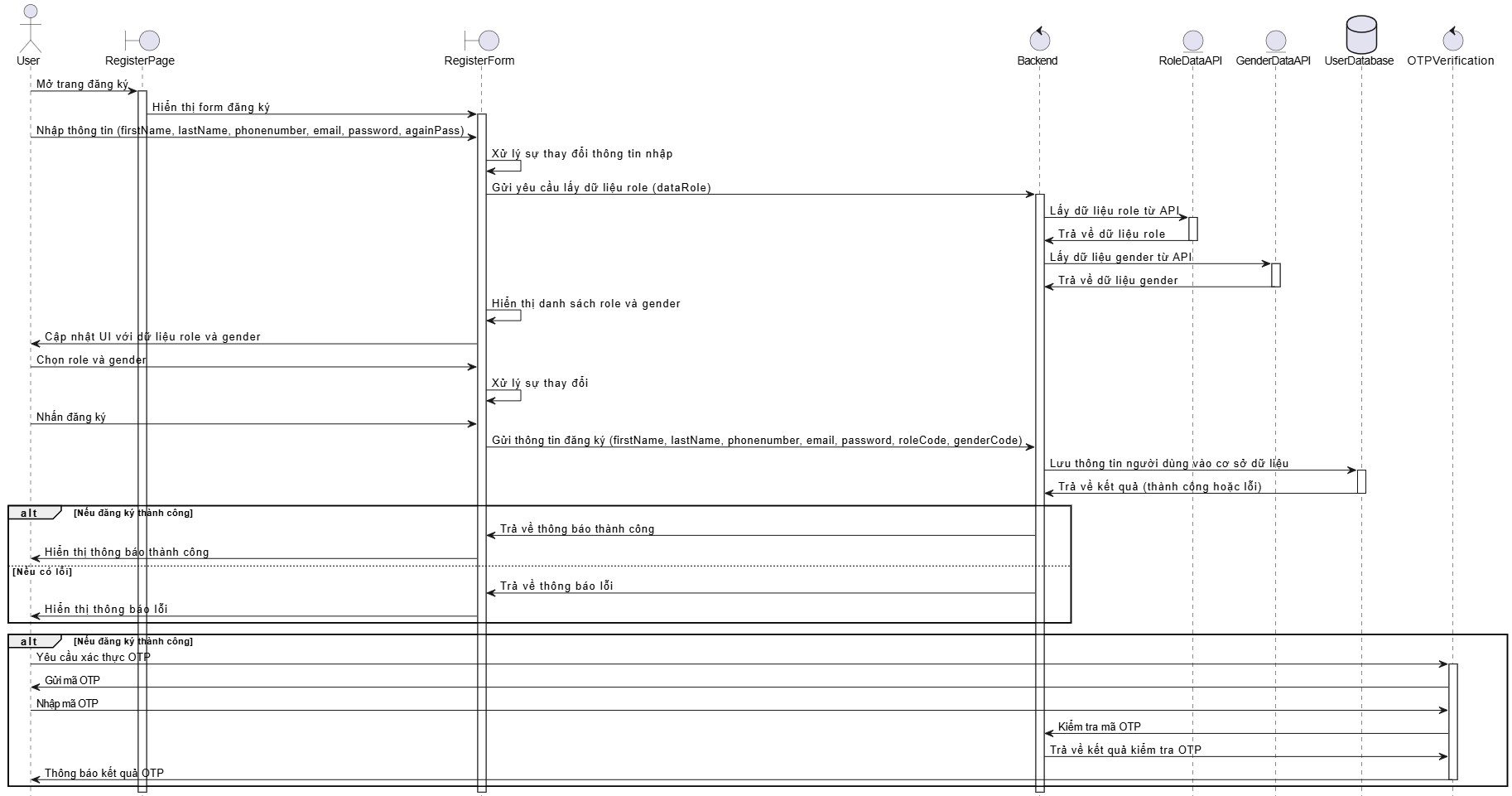
**Lược đồ Usecase nhóm Nhà tuyển dụng**

#### 1.3.2. Đặc tả Usecase

##### Usecase Đăng ký

**Đặc tả usecase Đăng ký**

|  |  |
| --- | --- |
| **Name** | **Đăng ký** |
| Brief description | Người dùng (nhà tuyển dụng hoặc ứng viên) đăng ký tài khoản mới trên hệ thống, xác thực thông qua OTP. |
| Actor(s) | Nhà tuyển dụng, Ứng viên |
| Pre-conditions | Người dùng chưa có tài khoản và đang ở trang chủ. |
| Post-conditions | * Nếu đăng ký thành công: Người dùng nhận được thông báo xác thực OTP thành công, được tạo mới tài khoản, thông tin được lưu vào CSDL. * Nếu đăng ký không thành công: Thông báo lỗi và yêu cầu người dùng kiểm tra lại thông tin hoặc thực hiện lại quá trình xác thực OTP. |
| Main flow | 1. Người dùng ở **trang chủ** và chọn **nút "Đăng ký"** ở góc trên bên trái.   * Hệ thống chuyển đến giao diện đăng ký.  1. Người dùng nhập các thông tin cá nhân:  * Họ tên * Số điện thoại * Email * Mật khẩu * Vai trò: Nhà tuyển dụng hoặc Ứng viên * Giới tính: Nam hoặc Nữ  1. Người dùng nhấn nút **"Đăng ký"**.  * Hệ thống kiểm tra tính hợp lệ của thông tin.  1. Hệ thống chuyển đến trang **xác thực OTP** và gửi mã OTP 6 chữ số đến email của người dùng. 2. Người dùng nhập mã OTP:  * **Nếu OTP hợp lệ**:   + Hệ thống thông báo **"OTP xác thực thành công"**.   + Hệ thống thông báo **"Tạo tài khoản thành công"** và chuyển người dùng đến trang chủ. * **Nếu OTP không hợp lệ**:   + Hệ thống hiển thị thông báo lỗi **"Xác thực OTP thất bại"** và yêu cầu người dùng nhập lại OTP. * **Nếu người dùng không nhận được OTP**:   + Người dùng chọn nút **"Gửi lại OTP"**, và hệ thống gửi lại mã OTP. |
| Alternative flow | 3.1. **Thông tin không hợp lệ khi đăng ký**   * Hệ thống hiển thị thông báo lỗi (ví dụ: số điện thoại đã tồn tại, mật khẩu phải tối đa 6 ký tự,…) và yêu cầu người dùng chỉnh sửa thông tin. |
| Exception Flow: | 2.1. **Kết nối mạng thất bại**   * Hệ thống thông báo lỗi kết nối và yêu cầu thử lại. |

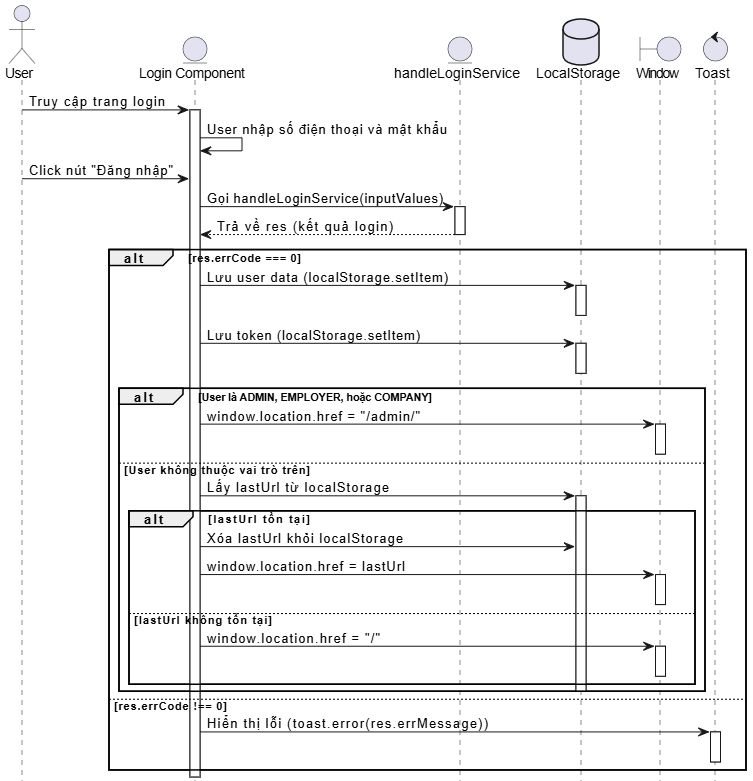


**Lược đồ sequence Đăng ký**

##### Usecase Đăng nhập

**Đặc tả usecase Đăng nhập**

|  |  |
| --- | --- |
| **Name** | **Đăng nhập** |
| Brief description | Người dùng (công ty, nhà tuyển dụng, ứng viên, quản trị viên) đăng nhập vào hệ thống. |
| Actor(s) | Công ty, Nhà tuyển dụng, Ứng viên, Quản trị viên |
| Pre-conditions | Người dùng đã có tài khoản và đang ở trang chủ. |
| Post-conditions | Người dùng đăng nhập thành công vào hệ thống hoặc nhận thông báo lỗi. |
| Main Flow | 1. Người dùng nhấn nút **Đăng nhập** ở góc trên bên phải trang chủ.   * Hệ thống hiển thị giao diện đăng nhập với hai trường: số điện thoại và mật khẩu.   2. Người dùng nhập **số điện thoại** và **mật khẩu**.   * Hệ thống xác nhận rằng các trường bắt buộc đều được điền.   3. Người dùng nhấn nút **Đăng nhập**.   * Hệ thống kiểm tra thông tin:   + Nếu đúng, người dùng được chuyển đến trang chủ.   + Hệ thống hiển thị trang chủ sau khi đăng nhập thành công. |
| Alternative flow | 3.1. **Thông tin đăng nhập không đúng**   * Hệ thống hiển thị thông báo: **"Số điện thoại hoặc mật khẩu không chính xác"**. * Người dùng quay lại bước 2 để nhập lại thông tin.   3.2. **Thiếu thông tin:**   * Hệ thống hiển thị thông báo: **"Missing required parameters"** (Thiếu thông tin bắt buộc). * Người dùng được yêu cầu điền đủ số điện thoại và mật khẩu. |
| Exception Flow | 2.1 **Kết nối mạng thất bại**   * Hệ thống hiển thị thông báo lỗi và yêu cầu thử lại. |

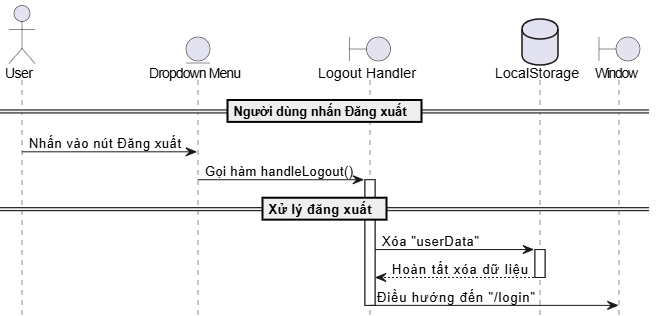


**Lược đồ sequence Đăng nhập**

##### Usecase Đăng xuất

**Đặc tả usecase Đăng xuất**

|  |  |
| --- | --- |
| **Name** | **Đăng xuất** |
| Brief description | Người dùng (công ty, nhà tuyển dụng, ứng viên, quản trị viên) đăng xuất khỏi hệ thống. |
| Actor(s) | Công ty, Nhà tuyển dụng, Ứng viên, Quản trị viên |
| Pre-conditions | * Người dùng đã đăng nhập vào hệ thống. * Người dùng ở bất kỳ trang nào trong hệ thống. |
| Post-conditions | Người dùng đăng xuất thành công. |
| Main Flow | 1. Người dùng nhấn vào tên của họ ở góc trên bên phải.   * Hệ thống hiển thị menu xổ ra với các tùy chọn.   2. Người dùng chọn **Đăng xuất** trong menu xổ ra.  3. Hệ thống thực hiện đăng xuất tài khoản của người dùng.   * Hệ thống chuyển người dùng đến trang đăng nhập. |
| Alternative flow | #N/A |
| Exception Flow | 2.1 **Kết nối mạng thất bại**   * Hệ thống hiển thị lỗi và yêu cầu thử lại. |

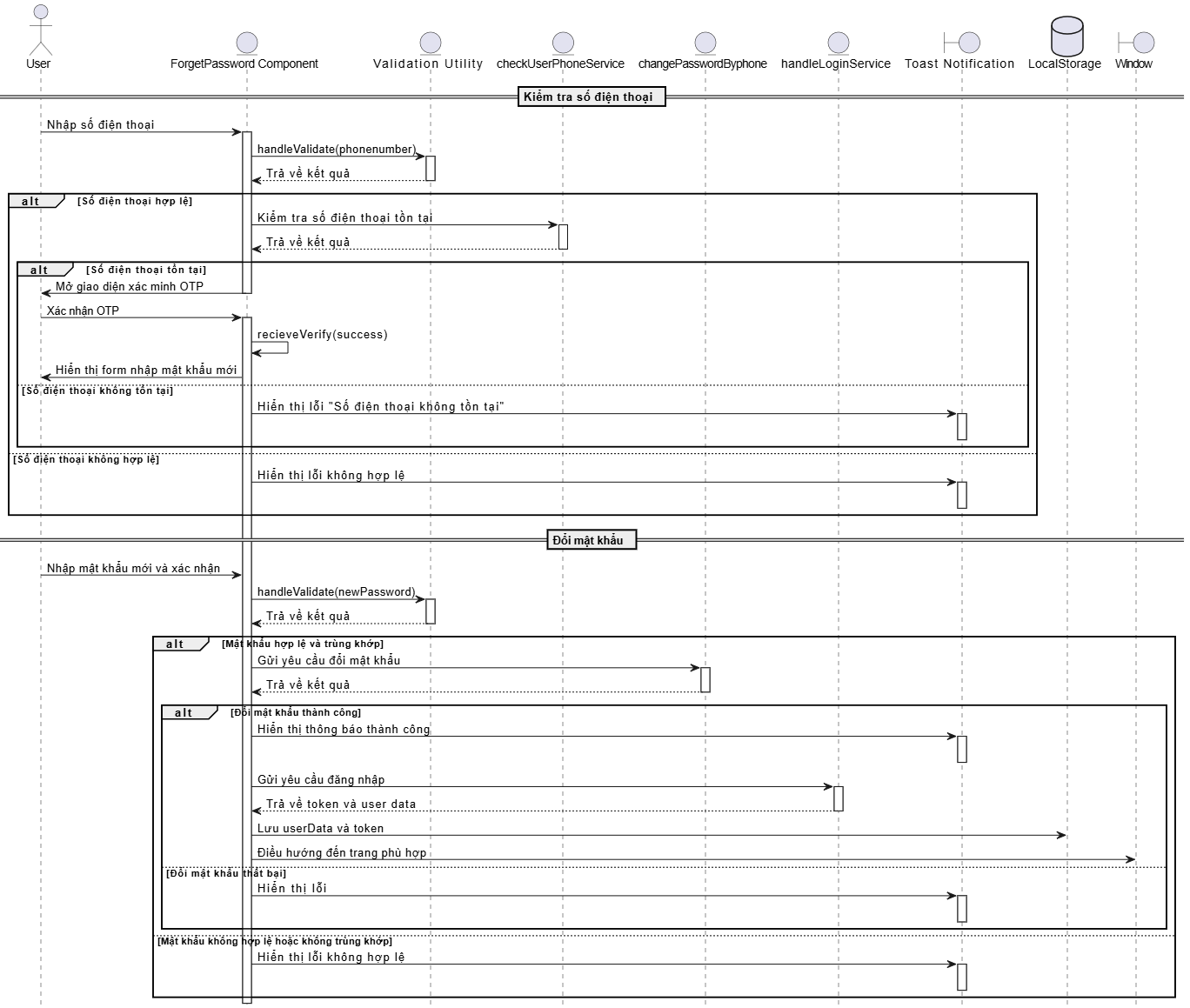


**Lược đồ sequence Đăng xuất**

##### Usecase Quên mật khẩu

**Đặc tả usecase Quên mật khẩu**

|  |  |
| --- | --- |
| **Name** | **Quên mật khẩu** |
| Brief description | Người dùng yêu cầu đặt lại mật khẩu khi quên mật khẩu hiện tại. |
| Actor(s) | Công ty, Nhà tuyển dụng, Ứng viên, Quản trị viên |
| Pre-conditions | Người dùng đã có tài khoản.  Người dùng đang ở trang Đăng nhập. |
| Post-conditions | Người dùng khôi phục mật khẩu thành công và được chuyển đến trang chủ.  Hoặc hệ thống hiển thị thông báo lỗi nếu có lỗi xảy ra. |
| Main flow | 1. Người dùng ở **Trang Đăng nhập**, nhấn vào nút **Quên mật khẩu**. **Hệ thống** hiển thị giao diện nhập **số điện thoại** đã đăng ký.  2. Người dùng nhập **số điện thoại** và nhấn nút **Xác nhận**. **Hệ thống** yêu cầu nhập **Email** liên kết với số điện thoại.  3. Người dùng nhập **Email** và nhấn nút **Gửi OTP**. **Hệ thống** gửi OTP đến địa chỉ email đã nhập và hiển thị thông báo: "OTP đã được gửi vào email của bạn."  4. Người dùng nhập mã **OTP** vào trường nhập OTP. **Hệ thống** kiểm tra OTP:   * Nếu OTP đúng, hệ thống hiển thị thông báo: "OTP xác thực thành công", chuyển đến bước 5. * Nếu OTP sai, hệ thống hiển thị thông báo: "Xác thực OTP thất bại", quay lại bước 4.   5. Người dùng nhập **Mật khẩu mới** và **Xác nhận mật khẩu**. **Hệ thống** kiểm tra:   * Nếu mật khẩu không trùng khớp, hiển thị thông báo: "Mật khẩu nhập lại không trùng khớp." * Nếu mật khẩu trùng khớp, hiển thị thông báo: "Đổi mật khẩu thành công", và chuyển người dùng đến **Trang chủ**. |
| Alternative flow | 3.1. Nếu người dùng không nhận được OTP sau một khoảng thời gian nhất định:   * **Hệ thống** hiển thị nút **Gửi lại OTP**. * Người dùng nhấn **Gửi lại OTP** để yêu cầu hệ thống gửi lại mã OTP. |
| Exception Flow | 2.1. Nếu số điện thoại chưa được đăng ký:   * **Hệ thống** hiển thị thông báo: "Số điện thoại không tồn tại” * Người dùng quay lại bước 1 để nhập lại thông tin. |

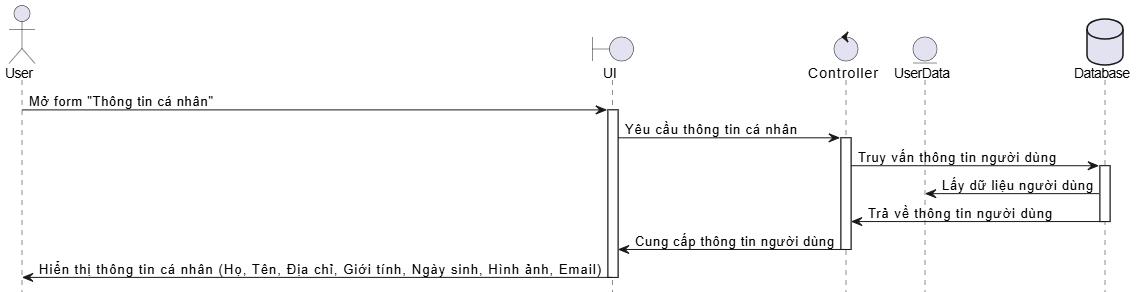


**Lược đồ sequence Quên mật khẩu**

##### Usecase Xem thông tin cá nhân

**Đặc tả usecase Xem thông tin cá nhân**

|  |  |
| --- | --- |
| **Name** | **Xem thông tin cá nhân** |
| Brief description | Người dùng xem thông tin cá nhân hiện tại của mình. |
| Actor(s) | Công ty, Nhà tuyển dụng, Ứng viên, Quản trị viên |
| Pre-conditions | Người dùng đã đăng nhập vào hệ thống. |
| Post-conditions | Thông tin cá nhân của người dùng được hiển thị. |
| Main Flow | 1. Người dùng nhấn vào tên của họ ở góc trên bên phải.   * Hệ thống hiển thị menu xổ ra với các tùy chọn.   2. Người dùng chọn **Thông tin** từ menu xổ ra.   * Hệ thống truy xuất dữ liệu và hiển thị thông tin cá nhân hiện tại của người dùng.   3. Người dùng có thể xem toàn bộ thông tin cá nhân của mình |
| Alternative flow | #N/A |
| Exception Flow | 2.1 **Kết nối mạng thất bại**   * Hệ thống không thể tải thông tin cá nhân và hiển thị lỗi. |

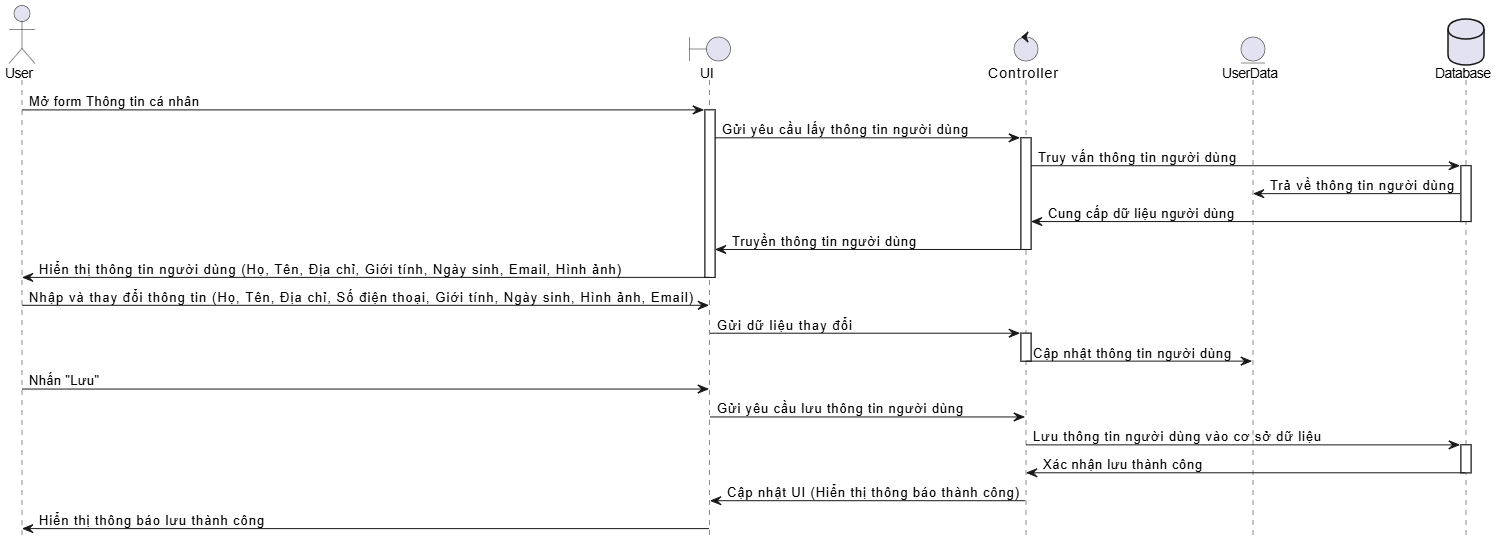


**Lược đồ sequence Xem thông tin cá nhân**

##### Usecase Cập nhật thông tin cá nhân

**Đặc tả usecase Cập nhật thông tin cá nhân**

|  |  |
| --- | --- |
| Name | Cập nhật thông tin cá nhân |
| Brief description | Người dùng thay đổi thông tin cá nhân của mình (họ tên, địa chỉ, hình ảnh, v.v.). |
| Actor(s) | Công ty, Nhà tuyển dụng, Ứng viên, Quản trị viên |
| Pre-conditions | Người dùng đã đăng nhập vào hệ thống. |
| Post-conditions | Thông tin cá nhân của người dùng được cập nhật thành công trong hệ thống. |
| Main Flow | **1. Người dùng nhấn vào tên của họ** ở góc trên bên phải màn hình.   * Hệ thống hiển thị menu xổ ra với các tùy chọn.   **2. Người dùng chọn "Thông tin cá nhân"** từ menu xổ ra.   * Hệ thống truy xuất dữ liệu và hiển thị thông tin cá nhân hiện tại của người dùng trên giao diện.   **3. Người dùng chỉnh sửa thông tin** tại các trường mong muốn, ví dụ:   * Họ tên * Email * Hình ảnh * Địa chỉ, v.v.   **4. Người dùng nhấn nút "Lưu."**  **5. Hệ thống cập nhật thông tin** vào cơ sở dữ liệu.  **6. Hệ thống hiển thị thông báo:**   * **"Cập nhật thông tin thành công."** |
| Alternative flow |  |
| Exception Flow | 5.1. **Dữ liệu không hợp lệ:**   * Hệ thống phát hiện thông tin không đúng định dạng (ví dụ: Email không hợp lệ). * Hiển thị thông báo lỗi: **"Thông tin không hợp lệ. Vui lòng kiểm tra lại."** |

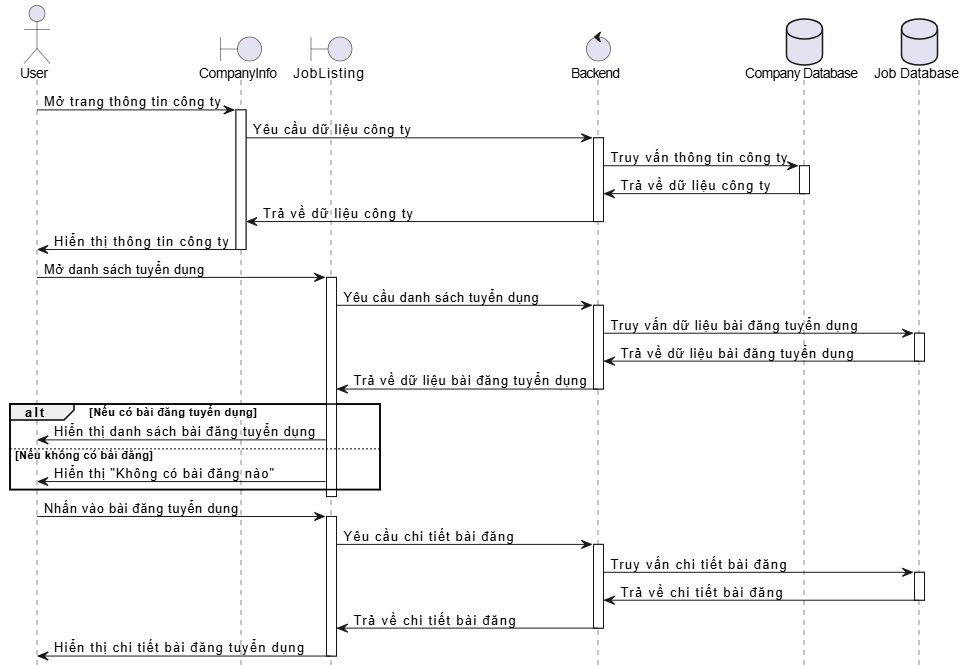


**Lược đồ sequence Cập nhật thông tin cá nhân**

##### Usecase Xem thông tin công ty

**Đặc tả usecase Xem thông tin công ty**

|  |  |
| --- | --- |
| **Name** | **Xem thông tin công ty** |
| Brief description | Ứng viên có thể xem danh sách các công ty có trên hệ thống và tìm kiếm công ty theo tên để xem thông tin chi tiết. |
| Actor(s) | Ứng viên |
| Pre-conditions |  |
| Post-conditions | Danh sách công ty hoặc thông tin chi tiết của một công ty được hiển thị. |
| Main flow | 1. Ứng viên truy cập trang chủ và nhấn vào **Công ty** ở đầu trang.   * Hệ thống hiển thị danh sách tất cả các công ty trên hệ thống.   2. Ứng viên có thể:   * Xem toàn bộ danh sách công ty. * Ứng viên nhập tên công ty vào thanh tìm kiếm. * Hệ thống lọc danh sách và hiển thị các công ty phù hợp với từ khóa tìm kiếm.   3. Ứng viên nhấn vào tên công ty trong danh sách để xem thông tin chi tiết.  4. Hệ thống hiển thị trang chi tiết của công ty |
| Alternative flow |  |
| Exception Flow: |  |

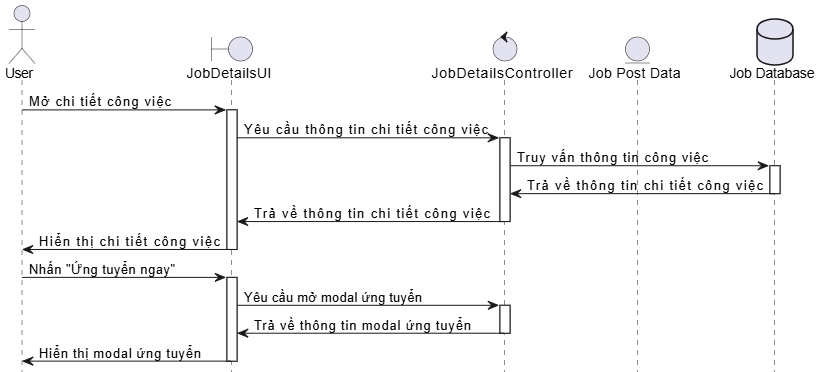


**Lược đồ sequence Xem thông tin công ty**

##### Usecase Xem bài tuyển dụng

**Đặc tả usecase Xem bài tuyển dụng**

|  |  |
| --- | --- |
| **Name** | **Xem bài tuyển dụng** |
| Brief description | Ứng viên có thể xem danh sách các bài đăng tuyển dụng, lọc kết quả theo các tiêu chí như lĩnh vực, hình thức làm việc, vị trí, và nhấp vào bài đăng để xem chi tiết thông tin tuyển dụng cũng như thông tin về công ty tuyển dụng. |
| Actor(s) | Ứng viên |
| Pre-conditions |  |
| Post-conditions | Danh sách các bài đăng tuyển dụng hoặc thông tin chi tiết của một bài đăng được hiển thị thành công. |
| Main Flow | 1. Người dùng truy cập trang chủ và nhấn vào mục **Việc làm** ở đầu trang.   * Hệ thống hiển thị danh sách tất cả các bài đăng tuyển dụng có trên hệ thống.   2. Người dùng có thể thực hiện một trong các hành động sau:   * **Xem toàn bộ danh sách các bài đăng tuyển dụng.** * **Sử dụng các tùy chọn lọc bên trái trang** để lọc danh sách bài đăng theo:   + **Lĩnh vực** (Ví dụ: Công nghệ thông tin, Kế toán, Marketing,...)   + **Hình thức làm việc** (Ví dụ: Toàn thời gian, Bán thời gian, Từ xa,...)   + **Vị trí làm việc** (Ví dụ: Hà Nội, TP. Hồ Chí Minh, Đà Nẵng,...)   + **Mức lương**, **Kinh nghiệm**, và các tiêu chí khác * Hệ thống cập nhật và hiển thị danh sách các bài đăng phù hợp với các tùy chọn lọc.   3. Người dùng nhấp vào một bài đăng tuyển dụng trong danh sách để xem chi tiết.   * Hệ thống hiển thị trang chi tiết của bài đăng tuyển dụng |
| Alternative flow | 2.1 Thay vì sử dụng bộ lọc, người dùng nhập từ khóa liên quan đến công việc vào **thanh tìm kiếm** ở đầu trang. |
| Exception Flow |  |

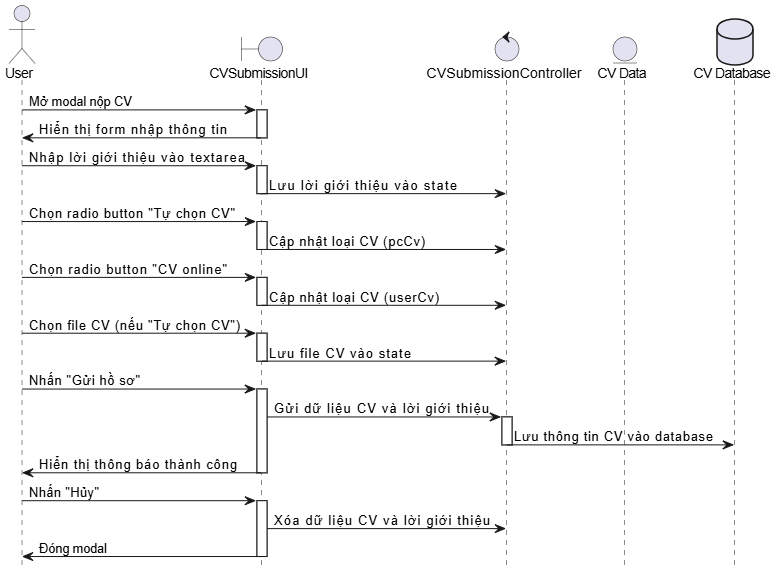


**Lược đồ sequence Xem bài tuyển dụng**

##### Usecase Ứng tuyển công việc

**Đặc tả usecase Ứng tuyển công việc**

|  |  |
| --- | --- |
| **Name** | **Ứng tuyển công việc** |
| Brief description | Ứng viên xem thông tin chi tiết của một bài tuyển dụng và nếu muốn ứng tuyển, họ có thể nhập lời giới thiệu bản thân và đính kèm CV từ máy hoặc CV online đã lưu trên hệ thống. Nếu thông tin đầy đủ, hệ thống sẽ gửi hồ sơ thành công. |
| Actor(s) | Ứng viên |
| Pre-conditions | Ứng viên đã đăng nhập vào hệ thống.  Ứng viên đã xem thông tin chi tiết của một bài tuyển dụng.  Nếu sử dụng CV online, ứng viên cần có CV đã lưu trên hệ thống. |
| Post-conditions | Hồ sơ của ứng viên được gửi thành công và lưu vào hệ thống.  Nếu ứng viên hủy ứng tuyển, hệ thống quay lại trang thông tin chi tiết bài tuyển dụng mà không lưu bất kỳ thông tin nào. |
| Main flow | 1. Ứng viên xem thông tin chi tiết của bài tuyển dụng và nhấn vào nút **Ứng tuyển ngay** trong mục thông tin công việc.  2. Hệ thống hiển thị giao diện ứng tuyển bao gồm:   * **Trường nhập lời giới thiệu bản thân.** * **Tùy chọn đính kèm CV từ máy tính** hoặc **chọn CV online** đã lưu trên hệ thống.   3. Ứng viên thực hiện một trong các hành động sau:   * **Nhập lời giới thiệu bản thân** vào trường thông tin. * **Đính kèm CV từ máy tính** hoặc chọn **CV online đã lưu trên hệ thống**.   4. Ứng viên nhấn vào nút **Gửi hồ sơ**:   * Nếu đầy đủ thông tin, hệ thống xử lý và thông báo: **"Đã gửi thành công."** * Nếu còn thiếu thông tin (ví dụ: chưa nhập lời giới thiệu hoặc chưa đính kèm CV), hệ thống thông báo: **"Gửi thất bại"** |
| Alternative flow | **3.1.** Nếu ứng viên chọn **CV online** nhưng chưa có CV nào được lưu trên hệ thống:   * Hệ thống hiển thị thông báo: **"Hiện chưa đăng CV online cho chúng tôi."** |
| Exception Flow: | **4.1.** Nếu hệ thống gặp lỗi khi gửi hồ sơ do vấn đề mạng hoặc lỗi server:   * Hệ thống hiển thị thông báo: **"Gửi thất bại."**   **4.2.** Nếu ứng viên nhấn vào nút **Hủy** thay vì gửi hồ sơ:   * Hệ thống thoát khỏi chức năng ứng tuyển và quay lại trang chi tiết bài tuyển dụng. |

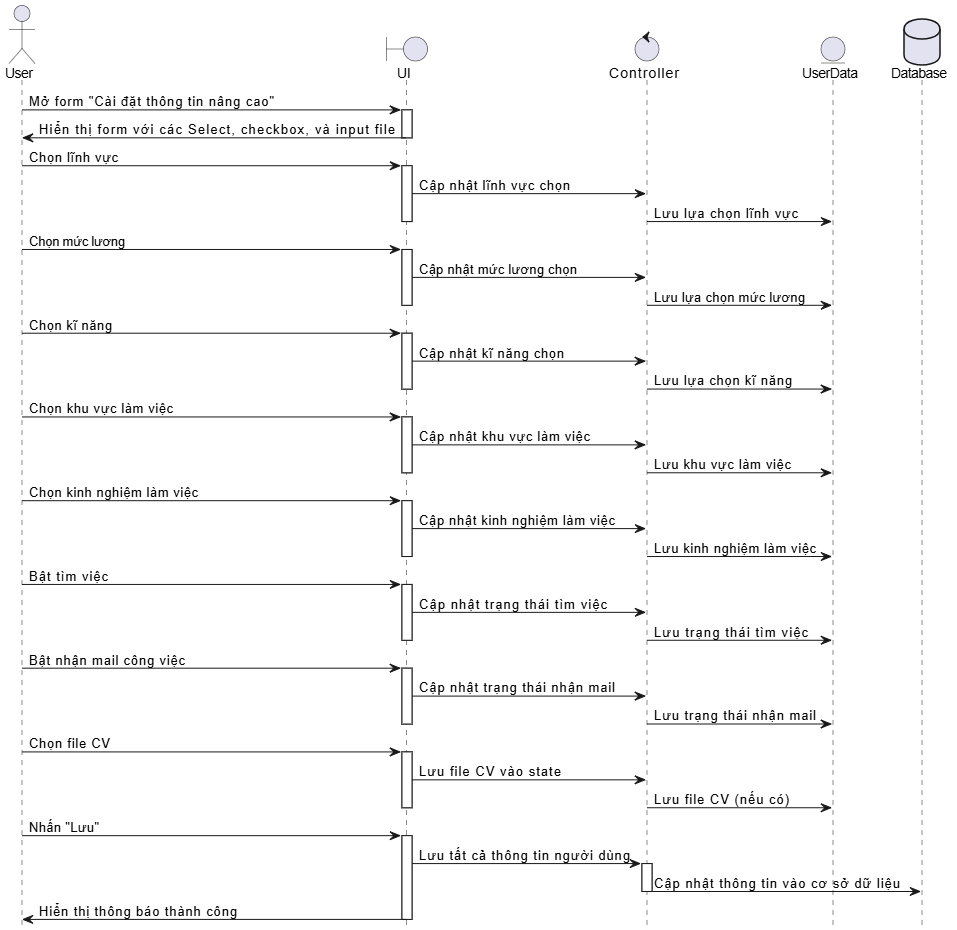


**Lược đồ sequence Ứng tuyển công việc**

##### Usecase Cài đặt nâng cao

**Đặc tả usecase Cài đặt nâng cao**

|  |  |
| --- | --- |
| **Name** | **Cài đặt nâng cao** |
| Brief description | Ứng viên truy cập vào mục **Cài đặt nâng cao** để cập nhật các thông tin cá nhân liên quan đến lĩnh vực làm việc, mức lương mong muốn, kỹ năng, khu vực làm việc, kinh nghiệm làm việc, và tùy chọn nhận thông báo công việc qua email. Ứng viên cũng có thể tải lên hoặc cập nhật CV từ máy tính để lưu trên hệ thống. |
| Actor(s) | Ứng viên |
| Pre-conditions | Ứng viên đã đăng nhập vào hệ thống. |
| Post-conditions | Thông tin cá nhân và CV được cập nhật thành công. |
| Main Flow | 1. Ứng viên nhấn vào **tên của mình** ở góc trên bên phải của giao diện.  2. Từ menu xổ ra, Ứng viên chọn **Cài đặt nâng cao**.  3. Hệ thống chuyển đến giao diện **Cài đặt nâng cao**, cho phép ứng viên cập nhật các thông tin sau:   * **Lĩnh vực làm việc**. * **Mức lương mong muốn**. * **Kỹ năng**. * **Khu vực làm việc**. * **Kinh nghiệm làm việc**. * **Tùy chọn bật/tắt tìm việc**. * **Tùy chọn nhận thông báo qua email**. * **CV**: Ứng viên chọn CV từ máy tính để lưu trên hệ thống.   4. Sau khi hoàn tất cập nhật, ứng viên nhấn vào nút **Lưu**.   * Hệ thống hiển thị thông báo: **"Cập nhật người dùng thành công."** |
| Alternative flow |  |
| Exception Flow |  |

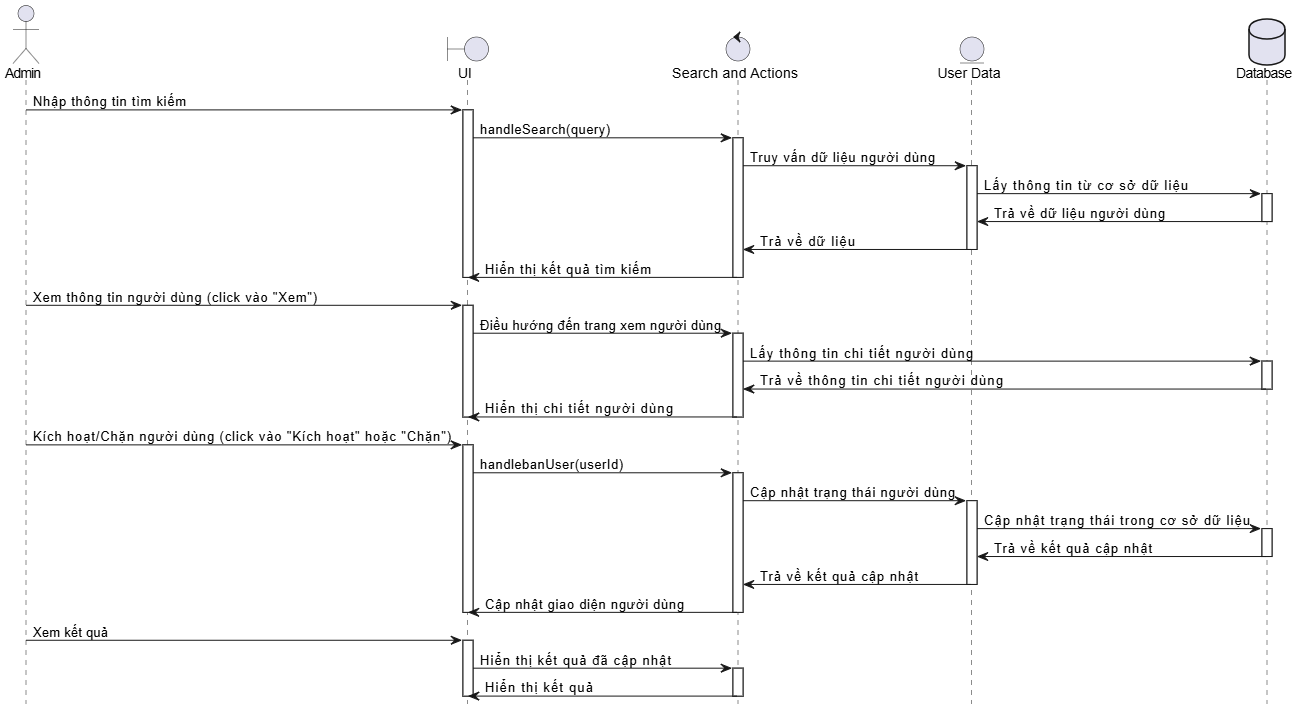


**Lược đồ sequence Cài đặt nâng cao**

##### Usecase Quản lý tài khoản

**Đặc tả usecase Quản lý tài khoản**

|  |  |
| --- | --- |
| **Name** | **Quản lý tài khoản** |
| Brief description | Admin đăng nhập vào hệ thống và truy cập vào trang quản trị. Tại trang quản trị, admin có thể quản lý tài khoản người dùng, bao gồm việc xem danh sách người dùng, cập nhật quyền, chặn tài khoản, hoặc thêm người dùng mới vào hệ thống. |
| Actor(s) | Admin |
| Pre-conditions | Admin đã đăng nhập vào hệ thống với quyền quản trị.  Admin đang ở trang quản trị hệ thống. |
| Post-conditions | Các thay đổi đối với tài khoản người dùng (cập nhật quyền, chặn tài khoản, thêm mới người dùng) được lưu vào hệ thống thành công.  Nếu admin không thực hiện thay đổi hoặc thoát, hệ thống không lưu bất kỳ thay đổi nào. |
| Main flow | 1. Admin đăng nhập vào hệ thống và vào trang quản trị.  2. Tại trang quản trị, admin chọn **Quản lý User** từ menu bên trái.  3. Hệ thống hiển thị hai lựa chọn:   * **Danh sách người dùng** * **Thêm người dùng**   4. **Nếu admin chọn "Danh sách người dùng"**:   * Hệ thống hiển thị danh sách tất cả các tài khoản đang hoạt động trên hệ thống. * Mỗi tài khoản có hai tùy chọn:   + **Xem**: Admin có thể xem thông tin chi tiết của tài khoản và cập nhật quyền cho tài khoản đó.   + **Chặn**: Admin có thể chặn tài khoản đó khỏi hệ thống.   5. **Nếu admin chọn "Thêm người dùng"**:   * Hệ thống hiển thị một form để admin nhập thông tin người dùng mới (tên, email, quyền, v.v.). * Admin nhập đầy đủ thông tin và nhấn nút **Lưu**. * Hệ thống báo: **"Thêm mới user thành công."** |
| Alternative flow |  |
| Exception Flow: |  |

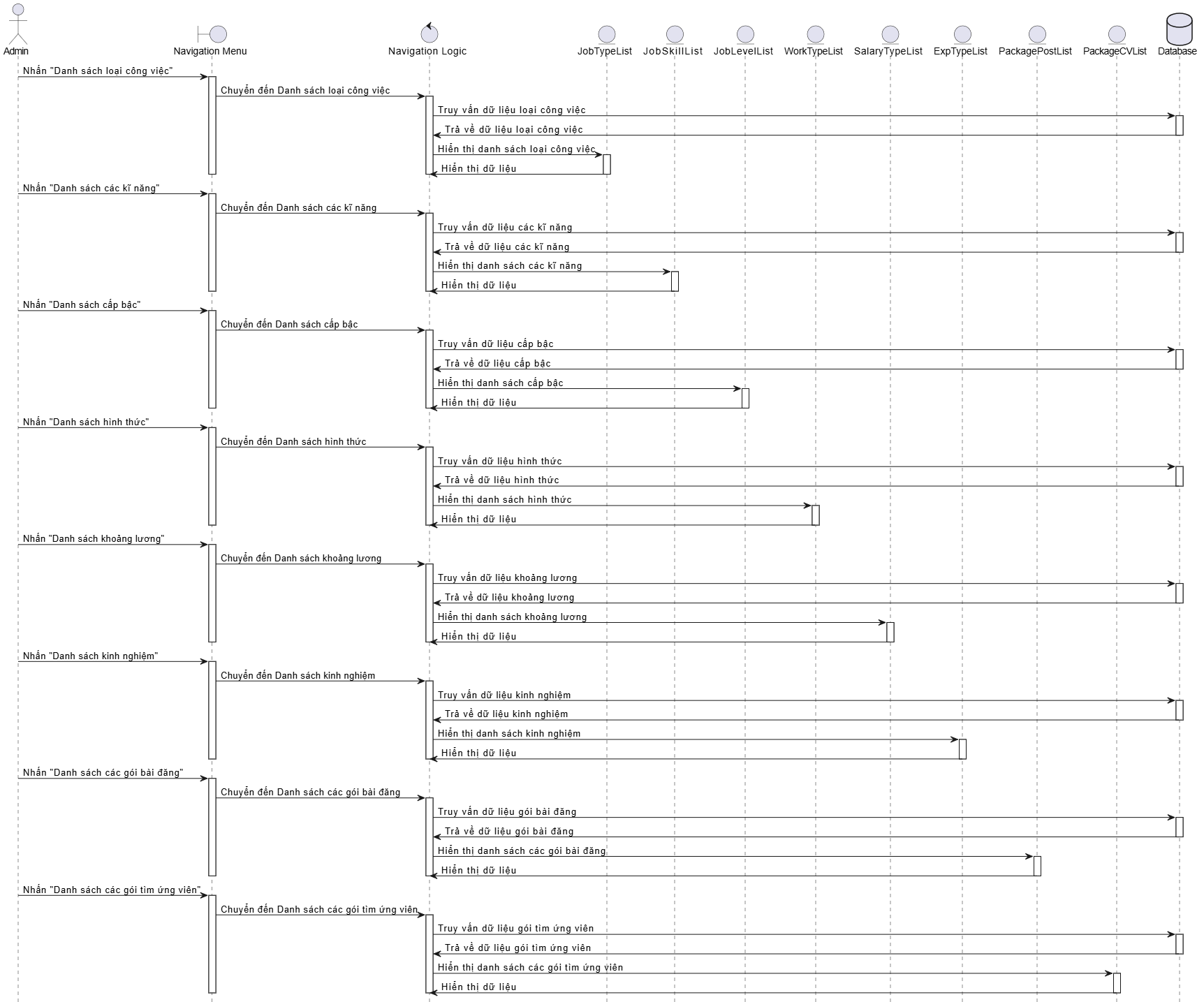


**Lược đồ sequence Quản lý tài khoản**

##### Usecase Quản lý danh mục hiển thị

**Đặc tả usecase Quản lý danh mục hiển thị**

|  |  |
| --- | --- |
| **Name** | **Quản lý danh mục hiển thị** |
| Brief description | Admin đăng nhập vào hệ thống và truy cập vào trang quản trị. Tại trang quản trị, admin có thể quản lý các danh mục hiển thị như loại công việc, kỹ năng, cấp bậc, hình thức làm việc, v.v. Mỗi danh mục có tùy chọn xem danh sách các mục hiện có hoặc thêm mục mới vào danh mục. | |
| Actor(s) | Admin | |
| Pre-conditions | Admin đã đăng nhập vào hệ thống với quyền quản trị.  Admin đang ở trang quản trị hệ thống. | |
| Post-conditions | Các thay đổi đối với danh mục hiển thị (thêm mới mục vào danh mục) được lưu vào hệ thống thành công.  Nếu admin không thực hiện thay đổi hoặc thoát, hệ thống không lưu bất kỳ thay đổi nào. | |
| Main Flow | 1. Admin đăng nhập vào hệ thống và vào trang quản trị.  2. Tại trang quản trị, admin chọn **Các danh mục hiển thị** từ menu bên trái.  3. Hệ thống hiển thị danh sách các danh mục hiển thị có sẵn trong hệ thống (bao gồm các danh mục như loại công việc, kỹ năng, cấp bậc, hình thức làm việc, v.v.).  4. **Nếu admin chọn một danh mục cụ thể**:   * Hệ thống hiển thị các mục trong danh mục đó (ví dụ: danh sách loại công việc, kỹ năng, cấp bậc, hình thức làm việc, v.v.). * Mỗi danh mục có hai tùy chọn:   + **Xem danh sách**: Xem chi tiết các mục hiện có trong danh mục đó.   + **Thêm mới**: Thêm mục mới vào danh mục đó.   5. **Nếu admin chọn "Thêm mới"**:   * Hệ thống hiển thị một form để admin nhập thông tin mới cho mục cần thêm. * Admin nhập đầy đủ thông tin và nhấn nút **Lưu**. * Hệ thống thông báo: **"Thêm mới danh mục thành công."** | |
| Alternative flow |  | |
| Exception Flow |  | |

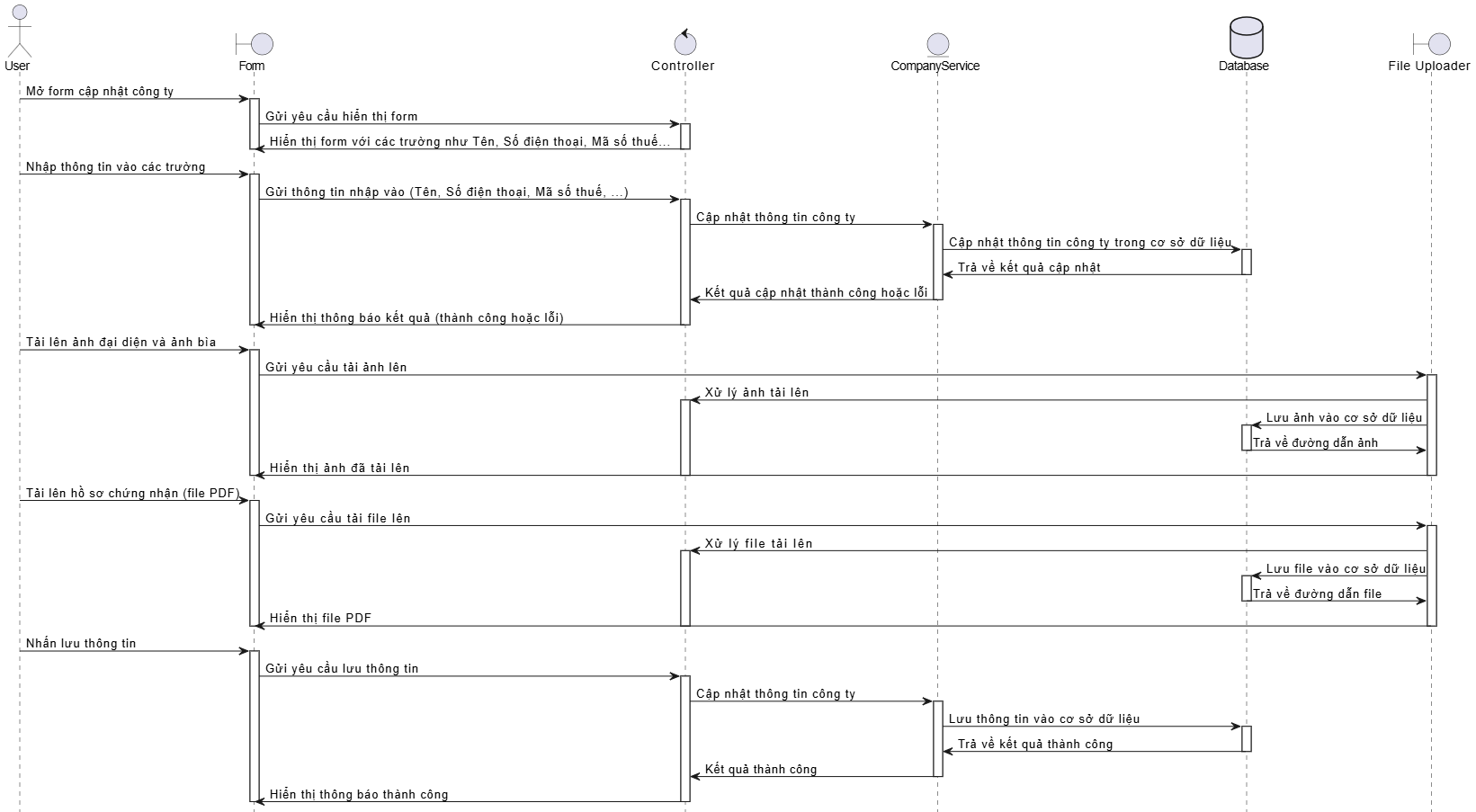


**Lược đồ sequence Quản lý danh mục hiển thị**

##### Usecase Quản lý công ty

**Đặc tả usecase Quản lý công ty**

|  |  |
| --- | --- |
| **Name** | **Quản lý công ty** |
| Brief description | Admin quản lý trạng thái hoạt động và xác thực của các công ty trên hệ thống. Admin có thể kích hoạt lại công ty, duyệt hoặc từ chối các công ty đang chờ kiểm duyệt, hoặc đưa công ty đã kiểm duyệt về trạng thái chờ kiểm duyệt. |
| Actor(s) | Admin |
| Pre-conditions | Admin đã đăng nhập vào hệ thống quản trị.  Danh sách các công ty hiển thị. |
| Post-conditions | Trạng thái của công ty được cập nhật thành công theo thao tác của admin.  Nếu admin hủy bỏ thao tác, hệ thống không thay đổi trạng thái công ty. |
| Main flow | 1. Admin chọn **Quản lý các công ty** từ menu bên trái.  2. Admin chọn **Danh sách các công ty.** Hệ thống hiển thị danh sách các công ty.  3. Admin thực hiện các thao tác sau:   * **Xem thông tin chi tiết công ty**: Admin nhấn vào nút xem chi tiết để hiển thị thông tin đầy đủ. * **Kích hoạt công ty** (đối với công ty không kích hoạt):   + Admin nhấn nút **Kích hoạt**.   + Hệ thống hiển thị thông báo xác nhận: **"Bạn có chắc muốn mở lại hoạt động công ty này?"**   + Nếu admin chọn **OK**, hệ thống cập nhật trạng thái công ty thành **Đã kích hoạt**. * **Duyệt công ty đang chờ kiểm duyệt**:   + Admin nhấn nút **Duyệt** để xác nhận công ty.   + Hệ thống hiển thị thông báo xác nhận: **"Bạn có chắc muốn duyệt công ty này?"**   + Nếu admin chọn **OK**, hệ thống cập nhật trạng thái thành **Đã kiểm duyệt** và thông báo: **"Đã duyệt công ty thành công."** * **Từ chối công ty đang chờ kiểm duyệt**:   + Admin nhấn nút **Từ chối**.   + Hệ thống cập nhật trạng thái thành **Không kích hoạt** và thông báo: **"Công ty đã bị từ chối."** * **Đưa công ty đã kiểm duyệt về trạng thái chờ kiểm duyệt**:   + Admin nhấn nút **Quay lại trạng thái chờ**.   + Hệ thống yêu cầu gửi lời nhắn đến nhà tuyển dụng.   + Sau khi gửi lời nhắn và nhấn hoành thành, hệ thống cập nhật trạng thái thành **Đang chờ kiểm duyệt**. |
| Alternative flow | 3.1. Nếu admin nhấn hủy trong quá trình xác nhận kích hoạt:   * Hệ thống không thay đổi trạng thái của công ty và quay lại giao diện danh sách công ty. |
| Exception Flow: | 3.2. Nếu hệ thống gặp lỗi (mạng, server) khi admin thực hiện thao tác:   * Trạng thái công ty không bị thay đổi. |

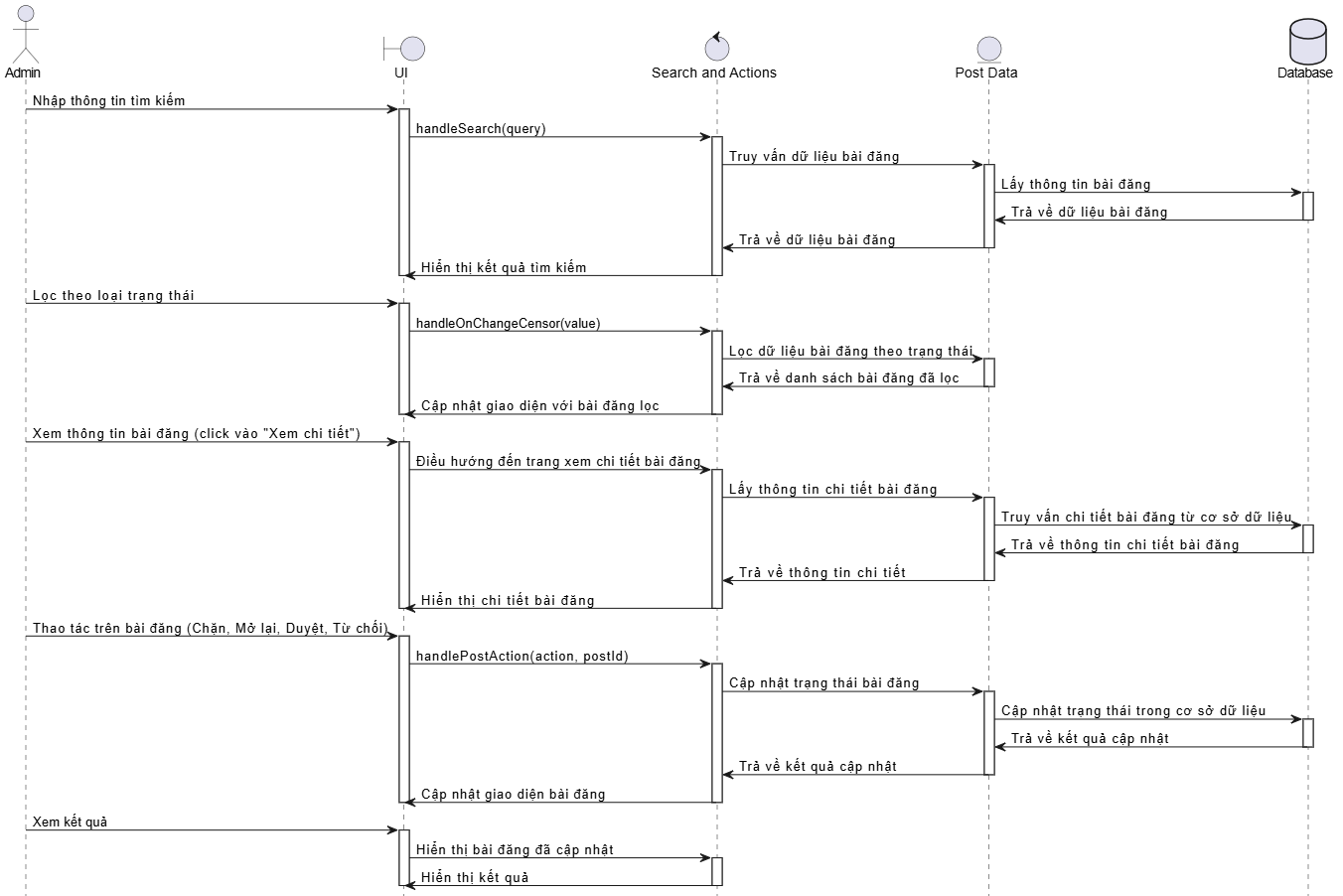


**Lược đồ sequence Quản lý công ty**

##### Usecase Quản lý bài đăng

**Đặc tả usecase Quản lý bài đăng**

|  |  |
| --- | --- |
| **Name** | **Quản lý bài đăng** |
| Brief description | Admin quản lý trạng thái của các bài đăng trong hệ thống. Admin có thể xem chi tiết bài đăng, duyệt, từ chối, chặn, hoặc mở lại bài đăng theo trạng thái cụ thể, với các yêu cầu nhập lời nhắn và xác nhận thao tác. |
| Actor(s) | Admin |
| Pre-conditions | Admin đã đăng nhập vào hệ thống quản trị.  Danh sách bài đăng được hiển thị đầy đủ. |
| Post-conditions | Trạng thái bài đăng được cập nhật thành công theo thao tác của admin.  Nếu admin hủy bỏ thao tác hoặc không nhập lời nhắn, trạng thái bài đăng không thay đổi. |
| Main Flow | 1. Admin vào mục **Quản lý bài đăng** từ menu bên trái.  2. Admin chọn **Danh sách bài đăng**, hệ thống hiển thị danh sách bài đăng.  3. Admin thực hiện một trong các thao tác sau:  **Xem chi tiết bài đăng**   * Admin bấm vào **Xem chi tiết** để xem thông tin bài đăng.   **Chặn bài đăng (Đối với bài đã kiểm duyệt)**   * Admin bấm nút **Chặn**. * Hệ thống yêu cầu nhập lời nhắn đến nhà tuyển dụng:   + Nếu không nhập lời nhắn, hệ thống hiển thị lỗi: **"Missing required parameters."**   + Nếu nhập lời nhắn và nhấn **Hoàn thành**, hệ thống hiển thị thông báo: **"Đã chặn bài viết thành công."**   **Mở lại bài đăng (Đối với bài đã bị chặn)**   * Admin bấm nút **Mở lại**. * Hệ thống yêu cầu nhập lời nhắn đến nhà tuyển dụng:   + Nếu không nhập lời nhắn, hệ thống hiển thị lỗi: **"Missing required parameters."**   + Nếu nhập lời nhắn và nhấn **Hoàn thành**, hệ thống hiển thị thông báo: **"Đã mở lại trạng thái chờ duyệt."**   **Duyệt bài đăng (Đối với bài chờ kiểm duyệt)**   * Admin bấm nút **Duyệt**. * Hệ thống hiển thị thông báo xác nhận: **"Bạn có chắc muốn duyệt bài viết này?"**   + Nếu nhấn **OK**, hệ thống hiển thị thông báo: **"Duyệt bài thành công."**   **Từ chối bài đăng (Đối với bài chờ kiểm duyệt)**   * Admin bấm nút **Từ chối**. * Hệ thống yêu cầu nhập lời nhắn đến nhà tuyển dụng:   + Nếu không nhập lời nhắn, hệ thống hiển thị lỗi: **"Missing required parameters."**   + Nếu nhập lời nhắn và nhấn **Hoàn thành**, hệ thống hiển thị thông báo: **"Đã từ chối bài thành công."** |
| Alternative flow | 3.1. Nếu admin nhấn hủy trong quá trình xác nhận kích hoạt:   * Hệ thống không thay đổi trạng thái của bài đăng và quay lại giao diện danh sách bài đăng. |
| Exception Flow | 3.2. Nếu hệ thống gặp lỗi (mạng, server) khi admin thực hiện thao tác:   * Trạng thái bài đăng không bị thay đổi. |

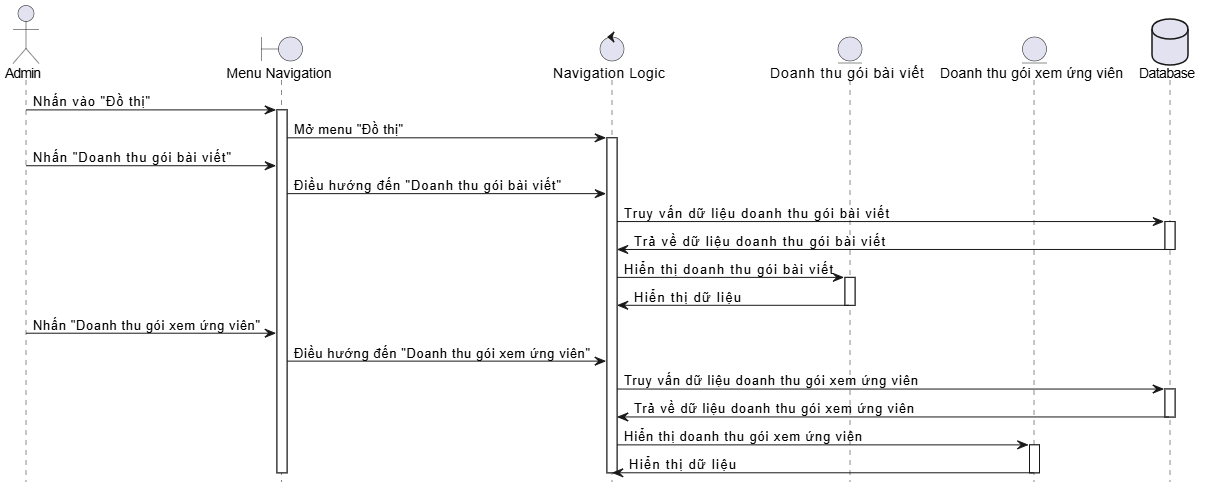


**Lược đồ sequence Quản lý bài đăng**

##### Usecase Xem báo cáo thống kê hệ thống

**Đặc tả usecase Xem báo cáo thống kê hệ thống**

|  |  |
| --- | --- |
| **Name** | **Xem báo cáo thống kê hệ thống** |
| Brief description | Admin có thể xem biểu đồ và bảng thống kê liên quan đến hoạt động hệ thống, bao gồm thống kê top lĩnh vực, doanh thu từ các gói bài đăng, và doanh thu từ các gói mua lượt xem ứng viên. Admin cũng có thể xuất dữ liệu thống kê ra file Excel. |
| Actor(s) | Admin |
| Pre-conditions | Admin đã đăng nhập vào hệ thống quản trị. |
| Post-conditions | Dữ liệu thống kê được hiển thị trên giao diện trang chủ.  File Excel được xuất thành công nếu admin thực hiện thao tác xuất. |
| Main flow | 1. Admin đăng nhập vào trang quản trị.  2. Admin chọn mục **Trang chủ** từ menu bên trái.  3. Hệ thống hiển thị các báo cáo thống kê bao gồm:   * **Biểu đồ thống kê top lĩnh vực**. * **Bảng thống kê doanh thu các gói bài đăng**. * **Bảng thống kê doanh thu các gói mua lượt xem ứng viên**.   4. Admin thực hiện một trong các thao tác sau:   * **Xem dữ liệu thống kê**:   + Dữ liệu được hiển thị trực tiếp trên giao diện trang chủ. * **Xuất file Excel**:   + Admin nhấn nút **Xuất Excel** ở mỗi bảng thống kê.   + Hệ thống xử lý và cung cấp file Excel để tải về. |
| Alternative flow | #NA |
| Exception Flow: | #NA |

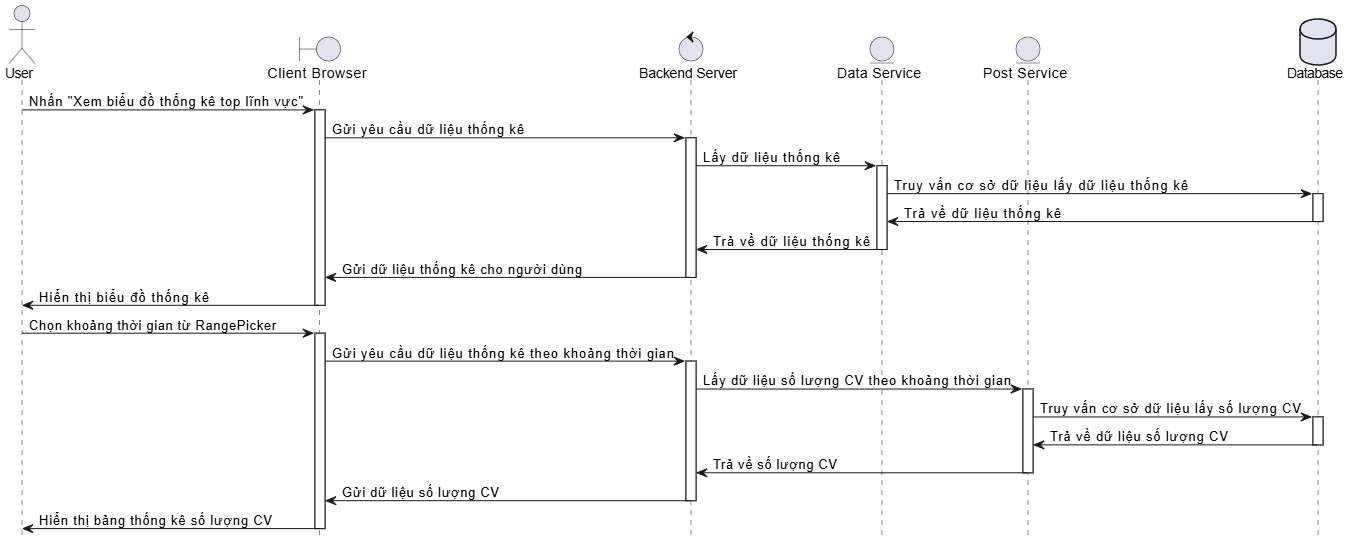


**Lược đồ sequence Xem báo cáo thống kê hệ thống**

##### Usecase Xem báo cáo thống kê công ty

**Đặc tả usecase Xem báo cáo thống kê công ty**

|  |  |
| --- | --- |
| **Name** | **Xem báo cáo thống kê công ty** |
| Brief description | Người dùng có thể xem các báo cáo thống kê liên quan đến hoạt động tuyển dụng của mình, bao gồm biểu đồ thống kê top lĩnh vực và bảng thống kê số lượng CV. |
| Actor(s) | Công ty  Nhà tuyển dụng |
| Pre-conditions | Người dùng đã đăng nhập vào hệ thống. |
| Post-conditions | Dữ liệu thống kê được hiển thị trên giao diện trang chủ. |
| Main Flow | 1. Người dùng đăng nhập vào hệ thống.  2. Người dùng chọn mục **Trang chủ** từ menu bên trái.  3. Hệ thống hiển thị các báo cáo thống kê, bao gồm:   * **Biểu đồ thống kê top lĩnh vực**. * **Bảng thống kê số lượng CV**.   4. Người dùng xem trực tiếp các dữ liệu này trên giao diện trang chủ. |
| Alternative flow | #N/A |
| Exception Flow | 3.1. Nếu hệ thống không thể hiển thị dữ liệu do lỗi kỹ thuật:   * Giao diện vẫn hiển thị nhưng không có dữ liệu thống kê. |

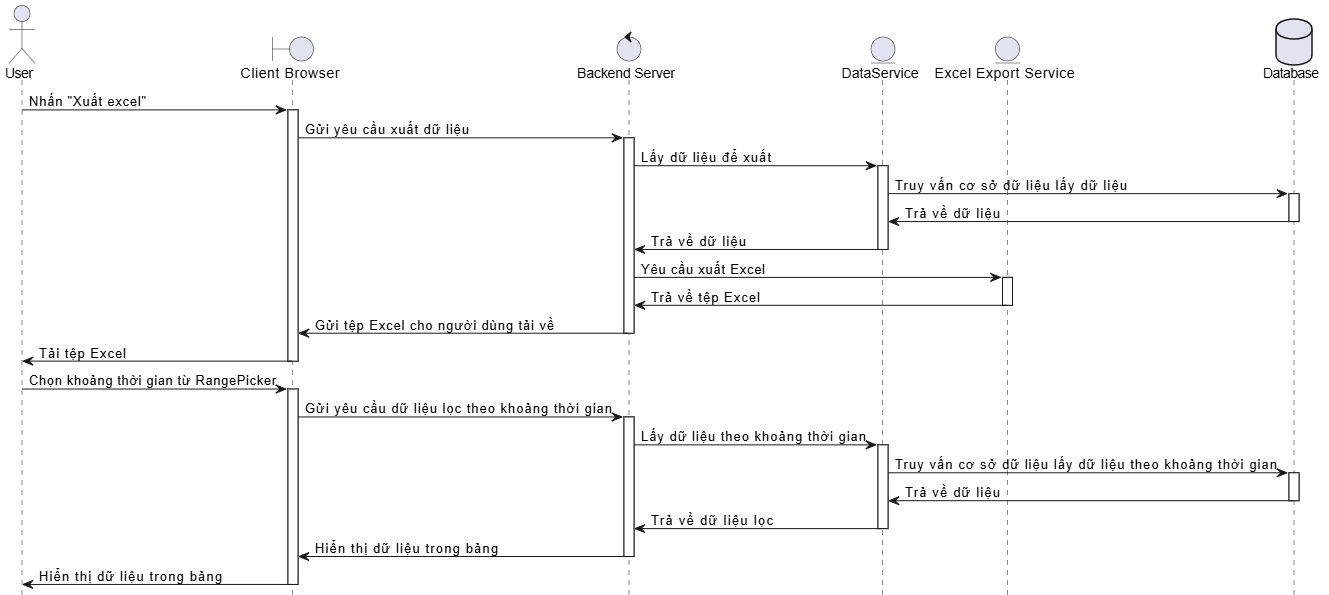


**Lược đồ sequence Xem báo cáo thống kê công ty**

##### Usecase Xem lịch sử giao dịch

**Đặc tả usecase Xem lịch sử giao dịch**

|  |  |
| --- | --- |
| **Name** | **Xem lịch sử giao dịch** |
| Brief description | Công ty có thể xem lịch sử giao dịch của mình, bao gồm lịch sử gói bài đăng và lịch sử gói xem ứng viên. Công ty cũng có thể xuất dữ liệu của từng bảng ra file Excel. |
| Actor(s) | Công ty |
| Pre-conditions | Công ty đã đăng nhập vào hệ thống. |
| Post-conditions | Lịch sử giao dịch được hiển thị trên giao diện.  File Excel được tải về máy nếu công ty thực hiện thao tác xuất. |
| Main flow | 1. Công ty đăng nhập vào hệ thống.  2. Công ty chọn mục **Lịch sử giao dịch** từ menu bên trái.  3. Hệ thống hiển thị hai bảng lịch sử giao dịch:   * **Lịch sử gói bài đăng**. * **Lịch sử gói xem ứng viên**.   4. Công ty thực hiện một trong các thao tác sau: 4.1. **Xem dữ liệu giao dịch**: - Dữ liệu hiển thị trực tiếp trên giao diện. 4.2. **Xuất file Excel**: - Công ty nhấn nút **Xuất Excel** ở mỗi bảng. - Hệ thống xử lý và cung cấp file Excel của bảng tương ứng để tải về. |
| Alternative flow | #N/A |
| Exception Flow: | 4.2.2. Nếu hệ thống không thể xuất file Excel do lỗi kỹ thuật:   * File không được xuất và giữ nguyên giao diện. |

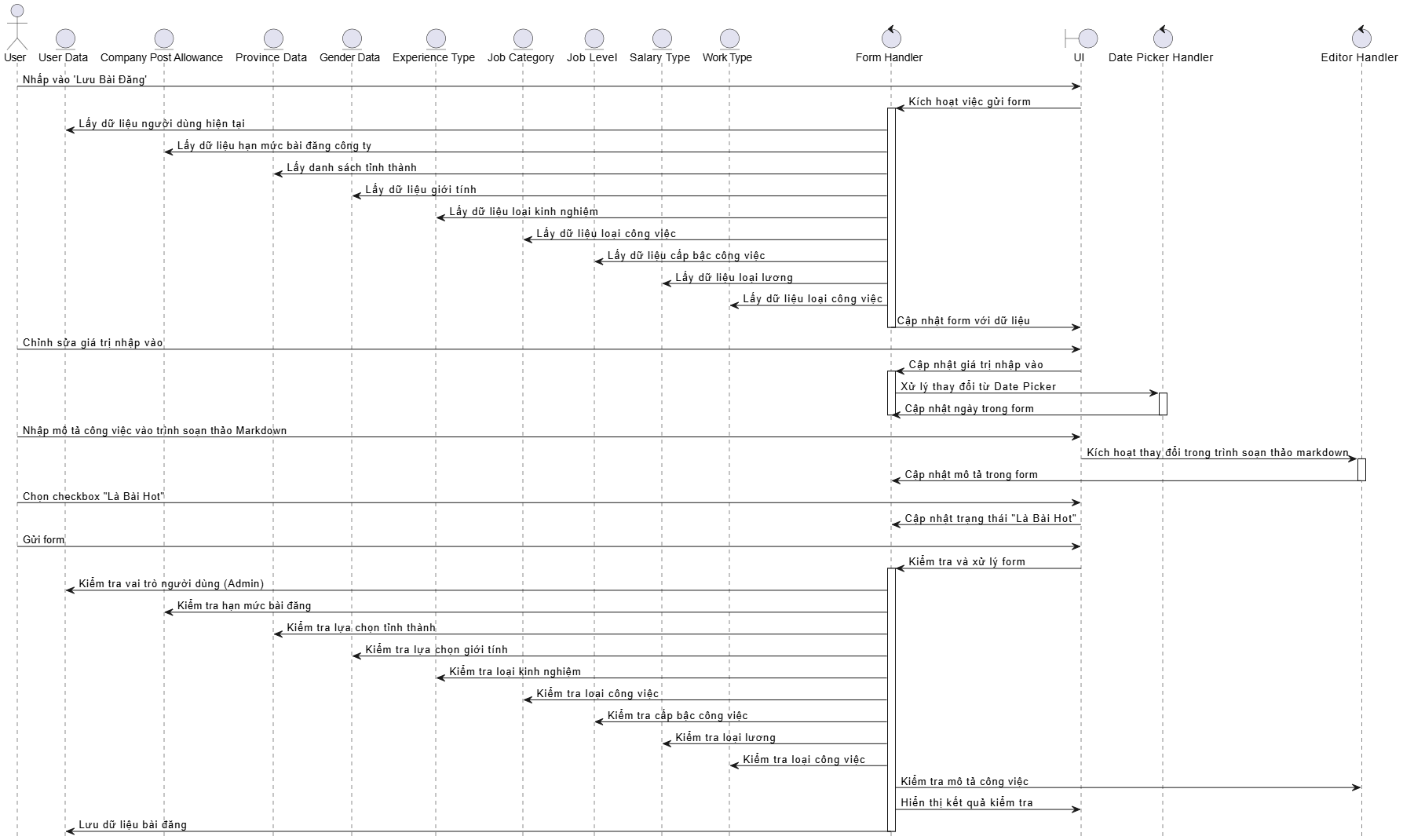


**Lược đồ sequence Xem lịch sử giao dịch**

##### Usecase Quản lý bài viết tuyển dụng

**Đặc tả usecase Quản lý bài viết tuyển dụng**

|  |  |
| --- | --- |
| **Name** | **Quản lý bài viết tuyển dụng** |
| Brief description | Công ty và nhà tuyển dụng có thể tạo bài đăng mới, xem danh sách bài đăng, xem hồ sơ ứng viên, và chỉnh sửa bài viết tuyển dụng qua mục Quản lý bài đăng. |
| Actor(s) | Công ty  Nhà tuyển dụng |
| Pre-conditions | Người dùng đã đăng nhập với vai trò **Công ty** hoặc **Nhà tuyển dụng**. |
| Post-conditions | Bài viết mới được thêm thành công và chuyển trạng thái chờ duyệt.  Bài viết được chỉnh sửa thành công và chuyển trạng thái chờ duyệt. |
| Main Flow | 1. Người dùng đăng nhập và chọn mục **Quản lý bài đăng** từ menu bên trái. 2. Hệ thống hiển thị hai tùy chọn:    * **Tạo mới bài đăng**.    * **Danh sách bài đăng**.   **Tạo mới bài đăng**   1. Người dùng chọn **Tạo mới bài đăng**. 2. Hệ thống hiển thị giao diện thêm thông tin bài đăng, bao gồm:    * Tên bài đăng.    * Địa chỉ.    * Số lượng nhân viên.    * Các thông tin cần thiết khác. 3. Người dùng nhập thông tin và nhấn nút **Lưu**: 5.1. Nếu còn thiếu thông tin, hệ thống hiển thị thông báo: **"Missing required parameters."** 5.2. Nếu thông tin đầy đủ, hệ thống hiển thị thông báo: **"Tạo bài viết tuyển dụng thành công. Hãy chờ quản trị viên duyệt."**   **Danh sách bài đăng**   1. Người dùng chọn **Danh sách bài đăng**. 2. Hệ thống hiển thị danh sách bài đăng với trạng thái tương ứng. 3. Người dùng thực hiện một trong các thao tác sau:    * **Xem CV nộp**:      + Người dùng bấm nút **Xem CV nộp** trong một bài đăng.      + Hệ thống hiển thị danh sách ứng viên đã ứng tuyển (Tên, Tỷ lệ phù hợp, Đánh giá,...)    * **Sửa bài đăng**:      + Người dùng bấm nút **Sửa** để chỉnh sửa thông tin bài đăng.      + Hệ thống hiển thị giao diện chỉnh sửa bài đăng.      + Người dùng chỉnh sửa thông tin và nhấn nút **Lưu**:        - Nếu còn thiếu thông tin, hệ thống hiển thị thông báo: **"Missing required parameters."**        - Nếu thông tin đầy đủ, hệ thống hiển thị thông báo: **"Đã chỉnh sửa bài viết thành công. Hãy chờ quản trị viên duyệt."** |
| Alternative flow | #N/A |
| Exception Flow | 5.1. Nếu hệ thống gặp lỗi kết nối khi lưu bài đăng mới:   * Hệ thống không thực hiện lưu bài đăng   8.2. Nếu hệ thống gặp lỗi khi chỉnh sửa bài đăng:   * Hệ thống không thực hiện chỉnh sửa bài đăng. |

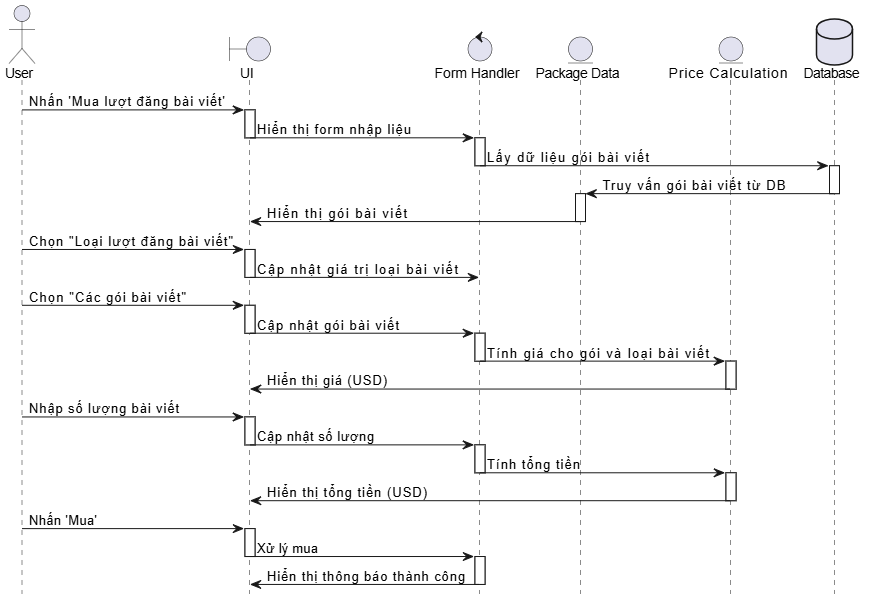


**Lược đồ sequence Quản lý bài viết tuyển dụng**

##### Usecase Mua thêm lượt đăng bài

**Đặc tả usecase Mua thêm lượt đăng bài**

|  |  |
| --- | --- |
| **Name** | **Mua thêm lượt đăng bài** |
| Brief description | Công ty thực hiện việc mua thêm lượt đăng bài thông qua giao diện Quản lý bài đăng. Công ty chọn loại bài viết, gói bài viết, nhập số lượng muốn mua và thực hiện thanh toán qua PayPal. |
| Actor(s) | Công ty |
| Pre-conditions | Công ty đã đăng nhập vào hệ thống.  Công ty đang ở giao diện Quản lý bài đăng trong menu bên trái. |
| Post-conditions | Lượt đăng bài đã được thêm vào tài khoản của công ty.  Nếu giao dịch thanh toán thất bại, không có lượt đăng bài nào được thêm vào. |
| Main flow | 1. Công ty truy cập mục **Quản lý bài đăng** từ menu bên trái.  2. Công ty nhấn vào tùy chọn **Mua thêm lượt đăng bài**.  3. Hệ thống hiển thị giao diện mua lượt đăng bài, bao gồm các thành phần:   * **Loại lượt đăng bài viết** (Danh sách lựa chọn: Ví dụ, "Bài viết thông thường," "Bài viết nổi bật"). * **Các gói bài viết** (Danh sách lựa chọn gói: Ví dụ, "Gói 5 lượt," "Gói 10 lượt"). * **Số lượng** (Ô nhập số lượng lượt bài viết muốn mua). * **Giá tiền** (Hiển thị tổng giá dựa trên lựa chọn của công ty).   4. Công ty thực hiện các hành động sau:   * Chọn loại lượt đăng bài viết. * Chọn gói bài viết phù hợp. * Nhập số lượng lượt đăng bài muốn mua.   5. Hệ thống hiển thị tổng giá tiền dựa trên thông tin đã nhập.  6. Công ty nhấn nút **Mua**.  7. Hệ thống chuyển công ty đến trang **PayPal** để thực hiện thanh toán. |
| Alternative flow | **6.1. Nếu công ty thay đổi thông tin trước khi nhấn nút mua:**   * Hệ thống cập nhật tổng giá tiền tương ứng. |
| Exception Flow: | **7.1. Nếu thanh toán qua PayPal thất bại:**   * Hệ thống hiển thị thông báo **"Thanh toán thất bại. Vui lòng thử lại."** * Hệ thống quay lại giao diện **Mua thêm lượt đăng bài**.   **7.2. Nếu công ty nhấn nút "Hủy" trên giao diện PayPal:**   * Hệ thống hủy giao dịch và quay lại trang **Quản lý bài đăng**, không có lượt đăng bài nào được thêm. |

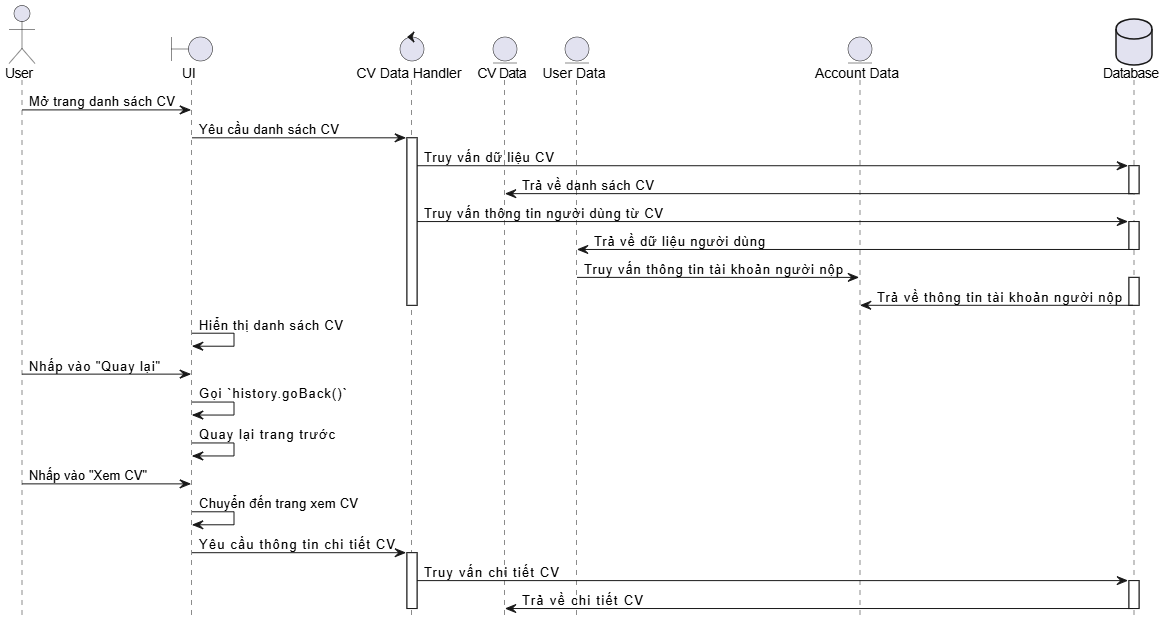


**Lược đồ sequence Mua thêm lượt đăng bài**

##### Usecase Xem danh sách ứng viên

**Đặc tả usecase Xem danh sách ứng viên**

|  |  |
| --- | --- |
| **Name** | **Xem danh sách ứng viên** |
| Brief description | Công ty hoặc nhà tuyển dụng truy cập danh sách ứng viên trong mục tìm kiếm ứng viên, lọc ứng viên theo các tiêu chí như lĩnh vực, kinh nghiệm, kỹ năng,... Sau khi danh sách được hiển thị, có thể chọn xem chi tiết từng ứng viên, bao gồm thông tin và CV. Mỗi lần xem chi tiết sẽ trừ một lượt xem. |
| Actor(s) | Công ty  Nhà tuyển dụng |
| Pre-conditions | Người dùng đã đăng nhập với quyền Công ty hoặc Nhà tuyển dụng.  Người dùng có số lượt xem ứng viên khả dụng. |
| Post-conditions | Số lượt xem ứng viên được trừ (nếu người dùng xem thông tin ứng viên chi tiết).  Không có thay đổi nếu người dùng hủy xem chi tiết ứng viên. |
| Main Flow | 1. Người dùng truy cập **Tìm kiếm ứng viên** từ menu bên trái.  2. Người dùng chọn mục **Danh sách ứng viên**.  3. Hệ thống hiển thị giao diện danh sách ứng viên với các bộ lọc, như lĩnh vực, kĩ năng, kinh nghiệm,…  4. Người dùng thực hiện lọc danh sách ứng viên bằng cách chọn hoặc nhập tiêu chí tìm kiếm.  5. Hệ thống hiển thị danh sách ứng viên với các thông tin sau:   * **STT** (Số thứ tự). * **Tên ứng viên**. * **Lĩnh vực**. * **Tỉ lệ phù hợp** * **Đánh giá**   6. Người dùng chọn một ứng viên và nhấn vào nút **Xem chi tiết**.  7. Hệ thống hiển thị thông báo: **"Khi xem bạn sẽ mất 1 lần xem thông tin ứng viên"**   * Nếu người dùng nhấn **OK**:   + Hệ thống trừ 1 lượt xem.   + Hiển thị chi tiết ứng viên, bao gồm thông tin và CV. * Nếu người dùng nhấn **Cancel**:   + Hệ thống quay lại danh sách ứng viên. |
| Alternative flow | **5.1. Nếu không có ứng viên nào phù hợp với tiêu chí lọc:**   * Hệ thống không hiển thị danh sách. |
| Exception Flow | **7.1. Nếu người dùng hết lượt xem ứng viên:**   * Hệ thống hiển thị thông báo: **"Công ty bạn đã hết lượt xem"** * Hệ thống quay lại danh sách ứng viên. |

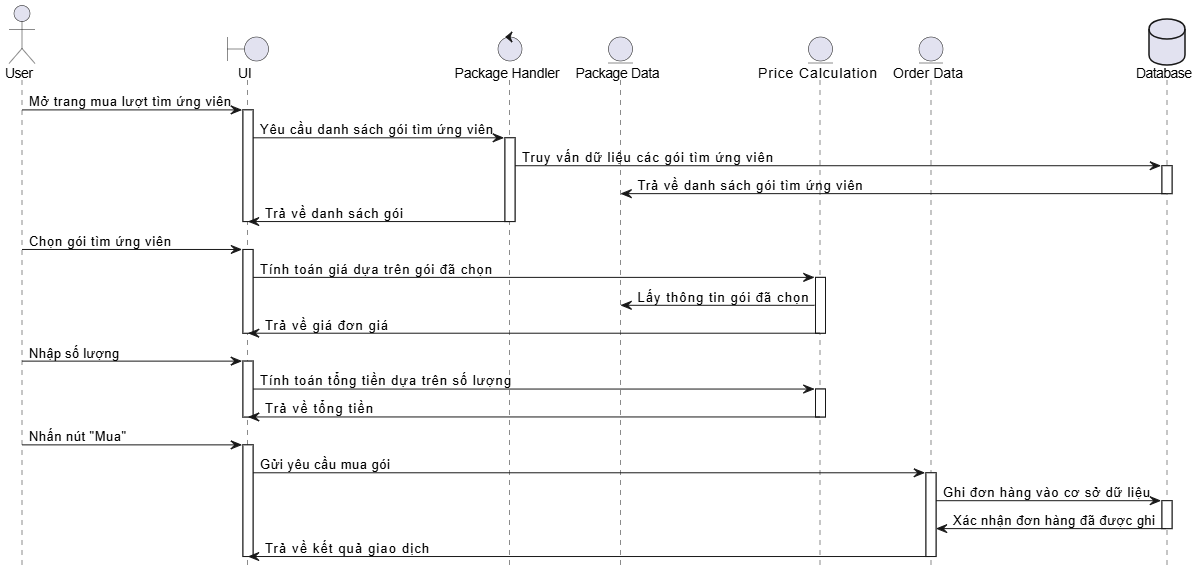


**Lược đồ sequence Xem danh sách Ứng viên**

##### Usecase Mua lượt xem ứng viên

**Đặc tả usecase Mua lượt xem ứng viên**

|  |  |
| --- | --- |
| **Name** | **Mua lượt xem ứng viên** |
| Brief description | Công ty truy cập mục **Mua thêm lượt xem ứng viên** trong **Tìm kiếm ứng viên**, chọn gói phù hợp, nhập số lượng lượt muốn mua và tiến hành thanh toán qua PayPal. |
| Actor(s) | Công ty |
| Pre-conditions | Công ty đã đăng nhập vào hệ thống.  Công ty đang ở giao diện Tìm kiếm ứng viên trong menu bên trái. |
| Post-conditions | Lượt xem ứng viên được thêm vào tài khoản của công ty.  Nếu thanh toán thất bại, không có lượt xem nào được thêm. |
| Main flow | 1. Công ty truy cập **Tìm kiếm ứng viên** từ menu bên trái.  2. Công ty chọn mục **Mua thêm lượt xem ứng viên**.  3. Hệ thống hiển thị giao diện mua lượt xem, bao gồm:   * **Các gói tìm ứng viên** (Danh sách gói với thông tin như "Gói 5 lượt," "Gói 10 lượt"). * **Số lượng** (Ô nhập số lượng lượt xem muốn mua). * **Giá tiền** (Hiển thị tổng giá tiền dựa trên gói và số lượng đã chọn).   4. Công ty thực hiện các bước sau:   * Chọn gói tìm ứng viên phù hợp. * Nhập số lượng lượt xem muốn mua. * Xem tổng giá tiền hiển thị.   5. Công ty nhấn nút **Mua**.  6. Hệ thống chuyển công ty đến trang thanh toán **PayPal**. |
| Alternative flow | **4.1. Nếu công ty thay đổi thông tin gói hoặc số lượng trước khi nhấn nút mua:**   * Hệ thống cập nhật lại tổng giá tiền tương ứng. |
| Exception Flow: | **6.1. Nếu thanh toán qua PayPal thất bại:**   * Hệ thống hiển thị thông báo: **"Thanh toán thất bại. Vui lòng thử lại."** * Hệ thống quay lại giao diện mua lượt xem. |

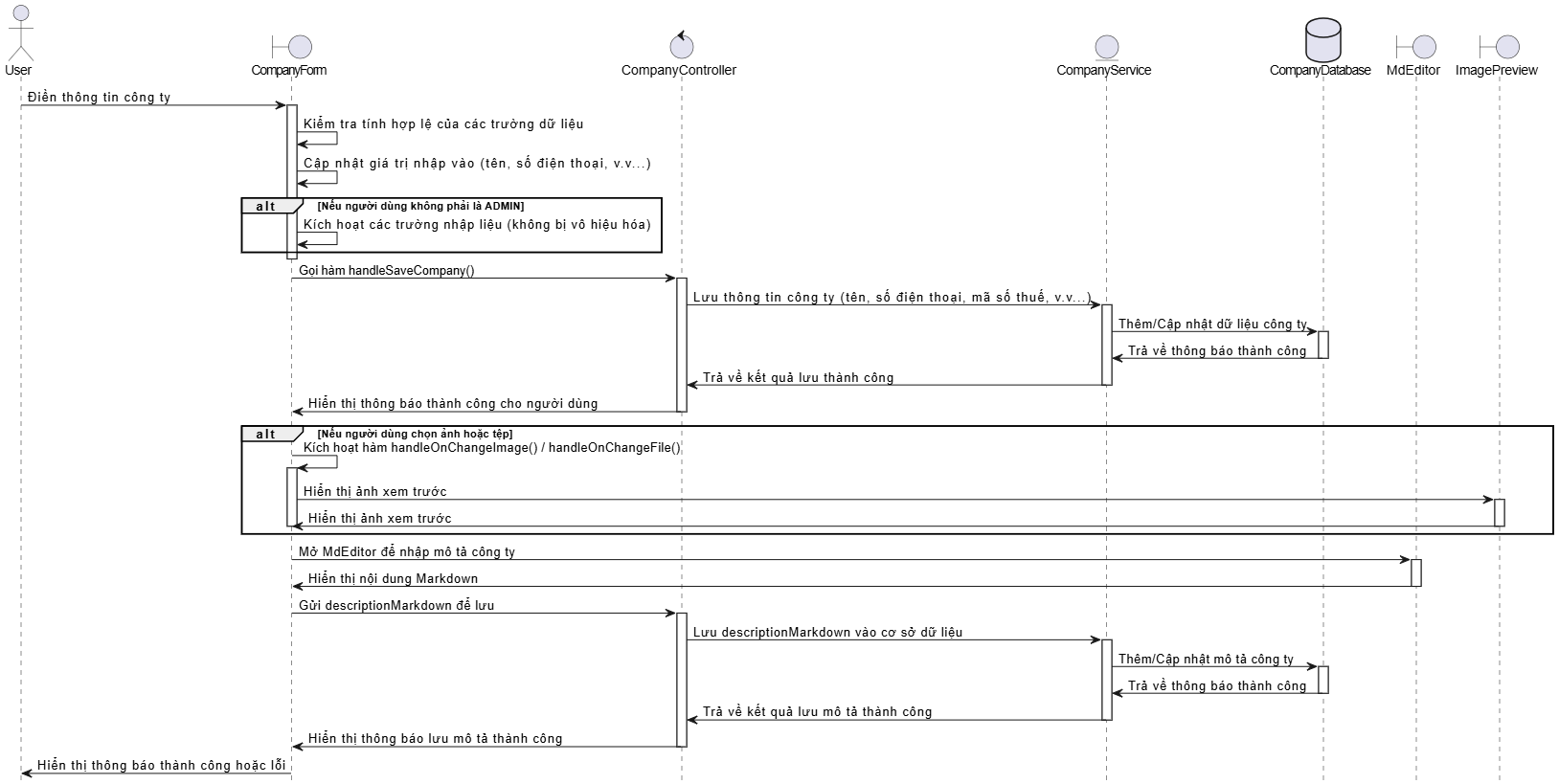


**Lược đồ sequence Mua lượt xem Ứng viên**

##### Usecase Tạo công ty

**Đặc tả usecase Tạo công ty**

|  |  |
| --- | --- |
| **Name** | **Tạo công ty** |
| Brief description | Nhà tuyển dụng chưa thuộc công ty nào có thể tạo mới một công ty bằng cách nhập thông tin đầy đủ và lưu lại. Nếu thông tin hợp lệ, công ty được tạo thành công và nhà tuyển dụng được cấp quyền quản trị công ty. |
| Actor(s) | Nhà tuyển dụng (chưa thuộc công ty nào) |
| Pre-conditions | Nhà tuyển dụng đã đăng nhập vào hệ thống.  Nhà tuyển dụng chưa thuộc bất kỳ công ty nào. |
| Post-conditions | Công ty mới được tạo và thông tin được lưu vào hệ thống.  Nhà tuyển dụng được cấp quyền quản trị công ty. |
| Main Flow | 1. Nhà tuyển dụng đăng nhập và truy cập **menu bên trái**.  2. Nhà tuyển dụng chọn mục **Công ty** và nhấn vào **Tạo mới công ty**.  3. Hệ thống hiển thị giao diện nhập thông tin công ty, bao gồm:   * **Tên công ty** * **Số điện thoại** * **Mã số thuế** * **Số nhân viên** * **Địa chỉ** * **Link website** * **Ảnh đại diện** * **Ảnh bìa** * **Hồ sơ chứng nhận** * **Giới thiệu công ty**   4. Nhà tuyển dụng điền đầy đủ thông tin và nhấn nút **Lưu**.  5. Hệ thống kiểm tra thông tin:   * Nếu thông tin đầy đủ và hợp lệ:   + Hệ thống lưu thông tin công ty và hiển thị thông báo: **"Tạo mới công ty thành công."**   + Nhà tuyển dụng được cấp quyền quản trị công ty. * Nếu thông tin thiếu:   + Hệ thống hiển thị thông báo: **"Missing required parameters."** |
| Alternative flow | #N/A |
| Exception Flow | #N/A |



**Lược đồ sequence Tạo công ty**

## **CHƯƠNG 2: KẾ HOẠCH KIỂM THỬ DỰ ÁN.**

### 2.1. Tổng quan.

#### 2.1.1. Mục đích.

Mục này được viết ra nhằm mô tả kế hoạch kiểm thử hệ thống Website đăng tin tuyển dụng và tìm việc làm có tự động sàng lọc CV phù hợp

Trong tài liệu này sẽ chỉ rõ mục tiêu, chiến lược, phương thức, kế hoạch, tiến độ và toàn bộ nội dung cần thiết cho hoạt động kiểm thử phần mềm dự án.

#### 2.1.2. Phạm vi.

Việc kiểm thử chia thành các giai đoạn khác nhau bao gồm:

* Integration Test (Kiểm thử tích hợp)
* System Test (Kiểm thử hệ thống)
* Acceptance Test (Kiểm thử chấp nhận)

#### 2.1.3. Giới thiệu về hệ thông được kiểm thử.

##### **2.1.3.1. Tổng quan về hệ thống.**

Website đăng tin tuyển dụng và tìm việc làm có tự động sàng lọc CV phù hợp được thiết kế nhằm hỗ trợ tối ưu trong việc kết nối giữa nhà tuyển dụng và ứng viên. Nền tảng này giúp các nhà tuyển dụng dễ dàng quản lý quy trình tuyển dụng, từ đăng tin, sàng lọc hồ sơ, đến thống kê và báo cáo hiệu quả. Đồng thời, ứng viên có thể nhanh chóng tìm kiếm cơ hội việc làm phù hợp dựa trên kỹ năng và kinh nghiệm cá nhân, mang lại sự tiện lợi và hiệu quả cho cả hai bên trong toàn quốc.

##### **2.1.3.2. Các chức năng xây dựng và nâng cấp hệ thống.**

**a. Người dùng chung**

**- Chức năng:**

* Đăng ký tài khoản.
* Đăng nhập/Đăng xuất.
* Quản lý thông tin cá nhân.
* Quên mật khẩu.

**b. Nhóm Ứng Viên**

**Chức năng:**

* **Tìm kiếm thông tin:**
  + Tìm kiếm bài viết tuyển dụng (chỉ bài đã được quản trị viên duyệt).
  + Tìm kiếm và xem thông tin các công ty đã đăng ký.
  + Xem trạng thái kiểm duyệt của công ty để đánh giá công ty.
* **Ứng tuyển:**
  + Nộp CV vào bài viết tuyển dụng phù hợp.
  + Xem lại danh sách CV đã nộp.
  + Xem chi tiết từng CV đã nộp.
* **Quản lý thông tin cá nhân:**
  + Cập nhật thông tin nâng cao (chia sẻ với nhà tuyển dụng khi cần).
  + Lưu trữ CV online, cho phép nhà tuyển dụng xem CV khi chia sẻ.
  + Chia sẻ thông tin cá nhân với nhà tuyển dụng.

**c. Nhóm Công Ty và Nhà Tuyển Dụng**

**Chức năng:**

* **Quản lý nhân sự:**
  + Nhóm công ty có thể:
    - Mời nhóm nhà tuyển dụng vào công ty.
    - Thôi việc nhóm nhà tuyển dụng trong công ty.
* **Quản lý bài viết:**
  + Đăng bài viết thuộc công ty (phụ thuộc số lượt đăng bài còn lại).
  + Quản lý thông tin bài viết của công ty.
* **Quản lý CV:**
  + Xem danh sách CV đã nộp vào từng bài viết.
  + Tìm kiếm và xem thông tin ứng viên theo nhu cầu.
* **Mua dịch vụ:**
  + Mua gói đăng bài tuyển dụng (bình thường và nổi bật).
  + Mua gói xem ứng viên.

**d. Quản Trị Viên (Admin)**

**Chức năng:**

* **Quản lý danh mục:**
  + Thêm, sửa, xóa thông tin các danh mục.
* **Quản lý tài khoản:**
  + Thêm, sửa, xóa tài khoản người dùng.
  + Kiểm duyệt thông tin đăng ký của công ty.
* **Quản lý bài viết:**
  + Duyệt, từ chối, chặn bài viết của công ty.
* **Quy định gói dịch vụ:**
  + Đặt số lượt, giá tiền, loại bài viết trong gói đăng bài.
  + Đặt số lượt và giá tiền cho gói xem ứng viên.

##### **2.1.3.3. Đặc điểm quá trình phát triển hệ thống.**

Hệ thống được xây dựng từ đầu, từ khâu lên ý tưởng đến thực hiện và kiểm thử.

##### **2.1.3.4. Trọng tâm nhiệm vụ kiểm thử.**

* **Người dùng chung:**
  + Đăng ký, đăng nhập/đăng xuất, và khôi phục mật khẩu hoạt động chính xác.
  + Đảm bảo cập nhật và bảo mật thông tin cá nhân.
* **Nhóm ứng viên:**
  + Tìm kiếm bài viết tuyển dụng được duyệt và thông tin công ty chính xác.
  + Nộp CV, quản lý danh sách CV đã nộp và tính năng chia sẻ CV hoạt động đúng.
* **Nhóm công ty và nhà tuyển dụng:**
  + Đăng và quản lý bài viết tuyển dụng theo số lượt đăng.
  + Xem và tìm kiếm thông tin ứng viên từ danh sách CV.
  + Quy trình mua và kích hoạt gói dịch vụ hoạt động ổn định.
* **Quản trị viên (Admin):**
  + Quản lý danh mục, tài khoản người dùng, và kiểm duyệt bài viết chính xác.
  + Đảm bảo thiết lập và áp dụng quy định gói dịch vụ đúng cách.

### 2.2. Chiến lược.

#### 2.2.1. Các kiểu kiểm thử.

Trong dự án này sẽ sử dụng các kiểu kiểm thử chính:

* + Kiểm Thử Đơn Vị (Unit Testing)
  + Kiểm Thử Hệ Thống (System Testing)
  + Kiểm Thử Chấp Nhận (Acceptance Testing)

#### 2.2.2. Giai đoạn kiểm thử.

**Giai đoạn 1: Kiểm Thử Đơn Vị (Unit Testing)**

* **Mục tiêu:**

Đảm bảo từng đơn vị nhỏ (module hoặc hàm) hoạt động đúng theo thiết kế.

* **Phương pháp:**
  + Lập đồ thị dòng điều khiển (Control Flow Graph).
  + Tính toán Cyclomatic Complexity để xác định số lượng test case cần thiết.
  + Thiết kế test case dựa trên các luồng điều khiển và dữ liệu.
  + Kiểm thử vòng đời biến để phát hiện các lỗi liên quan đến việc sử dụng biến.
    - **Kết quả mong đợi:**

Từng module hoặc hàm hoạt động đúng với yêu cầu và không phát hiện lỗi nghiêm trọng.

**Giai đoạn 2: Kiểm Thử Hệ Thống (System Testing)**

* + - **Mục tiêu:**

Đảm bảo toàn bộ hệ thống hoạt động chính xác khi các module được tích hợp hoàn chỉnh.

* **Phương pháp:**
  + Kiểm thử toàn bộ các chức năng chính như đăng ký, đăng nhập, tìm kiếm việc làm, và quản lý bài đăng tuyển dụng.
  + Kiểm thử giao diện người dùng (UI) để đảm bảo tính thân thiện và tương thích với các thiết bị.
  + Kiểm thử hiệu suất (Performance Testing) để đo khả năng chịu tải, stress, và xử lý dữ liệu lớn.
  + Kiểm thử bảo mật để phát hiện các lỗ hổng và bảo vệ dữ liệu người dùng.
* **Kết quả mong đợi:**

Hệ thống vận hành ổn định, đáp ứng yêu cầu về chức năng và hiệu suất.

**Giai đoạn 3: Kiểm Thử Chấp Nhận (Acceptance Testing)**

* **Mục tiêu:**

Đảm bảo hệ thống đáp ứng yêu cầu của người dùng cuối và sẵn sàng triển khai thực tế.

* **Phương pháp:**
  + Thực hiện kiểm thử chức năng từ góc độ người dùng cuối với các kịch bản thực tế.
  + Kiểm thử giao diện người dùng (UI Testing) để xác nhận tính dễ sử dụng và thân thiện.
  + Thực hiện kiểm thử Alpha: Kiểm tra bởi đội ngũ phát triển hoặc khách hàng trong môi trường giả lập.
* **Kết quả mong đợi:**

Hệ thống được xác nhận sẵn sàng triển khai và đáp ứng kỳ vọng của người dùng cuối.

#### 2.2.3. Công cụ kiểm thử.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mục đích** | **Tên công cụ** | **Nhà cung cấp** | **Phiên bản** |
| Thiết kế TestCase và lập Test Report | Excel | Microsoft | 2021 |
| Lập Test Plan | Word | Microsoft | 2021 |
| Ghi nhận và thực hiện kiểm thử tự động | Selenium | Open Source | 4.27.0 |

### 2.3. Tài nguyên.

#### 2.3.1. Nguồn lực.

|  |  |  |
| --- | --- | --- |
| **Vai trò** | **Số lượng** | **Trách nhiệm** |
| Nguyễn Hồng Thông Điệp (Test Leader) | 1 | * Lập kế hoạch kiểm thử chi tiết và định nghĩa chiến lược kiểm thử. * Phân công nhiệm vụ cụ thể cho các Tester trong nhóm. * Đảm bảo cung cấp đầy đủ các công cụ và tài nguyên cần thiết cho việc kiểm thử. * Theo dõi và giám sát tiến độ thực hiện kiểm thử. * Đánh giá mức độ hiệu quả của các Test Case và Test Script. * Lập báo cáo tổng kết chất lượng kiểm thử và đề xuất cải thiện quy trình kiểm thử. |
| Phạm Ngọc Đăng Khoa, Đoàn Thái Sơn (Testers), Trần Khải Hoàn (Tester) | 3 | * Thực hiện các kiểm thử theo kế hoạch và chiến lược đã định nghĩa. * Xây dựng và thực hiện các Test Case, Test Data, và Test Script. * Thực hiện kiểm thử tự động (nếu cần) và ghi nhận kết quả kiểm thử. * Xác định và báo cáo các lỗi (bugs) phát hiện trong hệ thống. * Hỗ trợ phân tích nguyên nhân lỗi và đề xuất các giải pháp khắc phục. |

#### 2.3.2. Môi trường.

Tiến hành kiểm thử trên Localhost.

### 2.4. TEST MILESTONES.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Đầu mục công việc** | **Nỗ lực**  **(manday/ man hours)** | **Ngày bắt đầu (dd/mm/yyyy)** | **Ngày kết thúc**  **(dd/mm/yyyy)** | **Nhân lực** |
| Lập kế hoạch kiểm thử | 7 ngày | 19/10/2024 | 25/10/2024 | Test leader |
| Trình bày các phương pháp kiểm thử đã áp dụng | 7 ngày | 2/11/2024 | 8/11/2024 | Tester |
| Chọn mã nguồn để thực hiện kiểm thử hộp trắng | 7 ngày | 9/11/2024 | 15/11/2024 | Tester |
| Dựa vào 8 kỹ thuật kiểm thử hộp đen để thiết kế test case | 14 ngày | 16/11/2024 | 29/11/2024 | Test leader |
| Lựa chọn và cài đặt một hệ thống quản lý lỗi | 7 ngày | 23/11/2024 | 29/11/2024 | Tester |
| Tổng hợp kết quả kiểm thử | 7 ngày | 30/11/2024 | 6/12/2024 | Test leader |

### 2.5. Tài liệu, báo cáo cần có.

Trong dự án, các tài liệu và báo cáo kiểm thử cần có bao gồm:

|  |  |  |
| --- | --- | --- |
| **Tên tài liệu** | **Mục đích** | **Người thực hiện** |
| Kế hoạch kiểm thử (Test Plan) | Đưa ra kế hoạch, chiến lược và các nội dung sẽ thực hiện để kiểm thử trong dự án | Test Leader |
| Thiết kế TestCase và TestData | Mô tả các TestCase và TestData cho từng chức năng của hệ thống | Tester và Test Leader |
| Kết quả từng lần | Kết quả từng lần được cập nhật trên sheet Kết quả test của file TestCase. | Tester |
| Báo cáo tổng hợp | Báo cáo tổng hợp kết quả test cho toàn hệ thống | Test Leader |

### 2.6. Đánh giá rủi ro.

Rủi ro: Lỗi phát sinh từ chức năng trên phiên bản cũ và do dữ liệu cũ gây ra.

Giải pháp: Retesr lại chức năng cũ và Chuẩn hóa lại dữ liệu cũ.

## **CHƯƠNG 3. CƠ SỞ LÝ THUYẾT VỀ KIỂM THỬ**

### 3.1. Các cấp độ kiểm thử

Tất cả các giai đoạn của quá trình phát triển phầm mềm đều trải qua quá trình kiểm thử phần mềm. Có 4 cấp độ kiểm thử phần mềm là:

* Kiểm thử đơn vị (Unit Testing)
* Kiểm thử tích hợp (Integration Testing)
* Kiểm thử hệ thống (System Testing)
* Kiểm thử chấp nhận (Acceptance Testing)

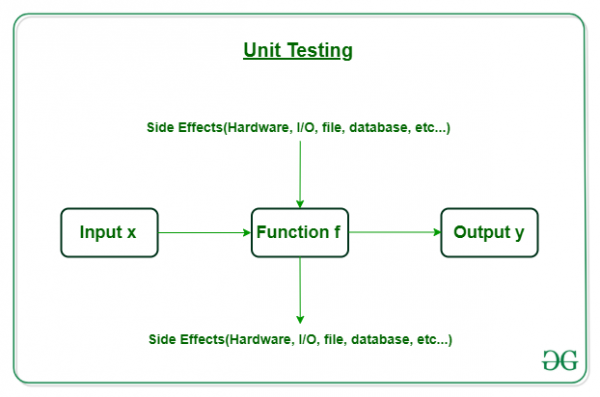
#### 3.1.1. Kiểm thử đơn vị

##### **3.1.1.1. Khái niệm**

Kiểm thử đơn vị (Unit Test) là giai đoạn kiểm thử từng lượng đơn vị (mô-đun) nhỏ trong hệ thống có hoạt động đúng hay chưa. Ở giai đoạn này, các Tester sẽ thực hiện kiểm thử riêng lẻ từng đơn vị trong 1 ứng dụng phần mềm. Đây là kiểu kiểm thử nằm trong phạm vi của kiểm thử hộp trắng (White Box Testing). Giai đoạn này thường được thực hiện bởi các Dev. Unit Test là giai đoạn kiểm thử đầu tiên được thực hiện trong quy trình kiểm thử một phần mềm bất kỳ. Hầu hết các lỗi được phát hiện ra trong giai đoạn này sẽ dễ sửa hơn so với các giai đoạn sau. Điều này giúp các Tester tiết kiệm được kha khá thời gian và công sức. Cũng giúp cho doanh nghiệp bớt tốn kém chi phí hơn nếu sản phẩm mắc phải những lỗi cơ bản. Ngoài ra, các bug được phát hiện ra trong giai đoạn này sẽ được sửa ngay mà không cần lưu lại hay quản lý như các giai đoạn khác.

##### **3.1.1.2. Quy trình thực hiện kiểm thử đơn vị**

Unit Test được thực hiện bằng cách sử dụng phương pháp Kiểm thử Hộp Trắng (White Box Test). Một quy trình Unit Testing thường bao gồm ba giai đoạn: Lập kế hoạch, dự trù các trường hợp có thể xảy ra, lên kịch bản và kiểm thử đơn vị. Trong bước đầu tiên, bài Unit Testing được chuẩn bị và xem xét. Bước tiếp theo là các trường hợp thử nghiệm và tập lệnh sẽ được thực hiện, sau đó code sẽ được thử nghiệm. Điều này yêu cầu các nhà phát triển trước tiên phải viết các bài kiểm thử đơn vị bị lỗi. Sau đó, họ viết mã và cấu trúc lại ứng dụng cho đến khi hoàn thành kiểm thử.



Sau khi tất cả các unit trong một chương trình được phát hiện là đang hoạt động theo cách hiệu quả nhất và không có lỗi, các thành phần lớn hơn của chương trình có thể được đánh giá bằng phương pháp kiểm thử tích hợp. Các bài Unit Testing nên được thực hiện thường xuyên và có thể được thực hiện thủ công hoặc có thể tự động. Dù có thể thực hiện thủ công, kiểm thử tự động là phương pháp phổ biến và nhanh chóng hơn. Các phương pháp tiếp cận tự động thường sử dụng khung kiểm thử để phát triển các trường hợp kiểm thử. Ngoài ra, kiểm thử tự động giúp dễ dàng gắn cờ và báo cáo mọi trường hợp kiểm thử không thành công đồng thời cung cấp bản tóm tắt các trường hợp kiểm thử.

##### **3.1.1.3. Vai trò của kiểm thử đơn vị trong kiểm thử phần mềm**

* Làm cho quy trình trở nên linh hoạt: Một trong những lợi ích chính của Unit Testing là nó làm cho quá trình mã hóa trở nên nhanh chóng hơn. Khi bạn thêm ngày càng nhiều tính năng vào một phần mềm, đôi khi bạn cần thay đổi lại những thiết kế và code cũ. Tuy nhiên, việc thay đổi code đã được kiểm thử sẽ dẫn tới nhiều rủi ro và tốn kém. Nếu bạn đã có sẵn các bài Unit Testing, bạn có thể tiến hành tái cấu trúc một cách dễ dàng.
* Cải thiện chất lượng code: Unit Testing xác định mọi khiếm khuyết có thể xuất hiện trước khi code được gửi thêm để kiểm thử tích hợp. Việc xây dựng các bài Unit Testing trước khi viết code thực tế sẽ khiến bạn xây dựng và cải thiện chất lượng code.
* Phát hiện sớm các lỗi (bug): Các vấn đề thường được tìm thấy ở giai đoạn đầu. Vì Unit Testing được thực hiện bởi các nhà phát triển, những người kiểm thử mã riêng lẻ trước khi tích hợp, các vấn đề có thể được phát hiện rất sớm và có thể được giải quyết ngay từ đó mà không ảnh hưởng đến các phần khác của code. Điều này bao gồm cả bug trong quá trình triển khai của dev và các sai sót hoặc thiếu các phần của đặc điểm kỹ thuật cho thiết bị.
* Tạo điều kiện cho những thay đổi và đơn giản hóa việc tích hợp: Unit Testing cho phép nhà phát triển cấu trúc lại code hoặc nâng cấp thư viện hệ thống sau đó và đảm bảo rằng module vẫn hoạt động chính xác. Unit Testing phát hiện những thay đổi có thể phá vỡ các kế hoạch ban đầu, làm giảm các khiếm khuyết trong các tính năng mới được phát triển hoặc giảm lỗi khi thay đổi chức năng hiện có. Ngoài ra, Unit Testing xác minh tính chính xác của từng unit. Sau đó, các unit được tích hợp vào một phần mềm bằng cách kiểm tra các phần của phần mềm thông qua Unit Testing. Việc kiểm thử phần mềm sau này trong quá trình tích hợp dễ dàng hơn do đã có sự xác minh của các unit riêng lẻ.
* Cung cấp tài liệu: Unit Testing cung cấp tài liệu về hệ thống. Các nhà phát triển muốn tìm hiểu chức năng nào được cung cấp bởi một unit và cách sử dụng nó có thể xem xét các bài Unit Testing để hiểu cơ bản về giao diện (API) của unit đó.
* Quy trình gỡ lỗi: Unit Testing giúp đơn giản hóa quá trình gỡ lỗi. Nếu kiểm thử không thành công, thì chỉ những thay đổi mới nhất được thực hiện trong code mới cần được gỡ lỗi.
* Giảm thiểu chi phí: Vì các bug được phát hiện sớm nên Unit Testing giúp giảm chi phí sửa lỗi. Các bug được phát hiện sớm hơn sẽ dễ sửa hơn vì các bug được phát hiện muộn hơn thường là kết quả của nhiều thay đổi và bạn không thực sự biết bug nào đã gây ra lỗi. Chi phí sẽ bị đội lên rất nhiều để tìm kiếm ngọn nguồn nguyên nhân.

##### **3.1.1.4. Nhược điểm kiểm thử đơn vị**

* Phạm vi hạn chế: Chỉ kiểm tra từng module nhỏ nên khó phát hiện lỗi liên quan đến tích hợp giữa các module.
* Phụ thuộc mock/stub: Dữ liệu giả (mock/stub) có thể không phản ánh đúng hành vi thực tế.

#### 3.1.2. Kiểm thử tích hợp

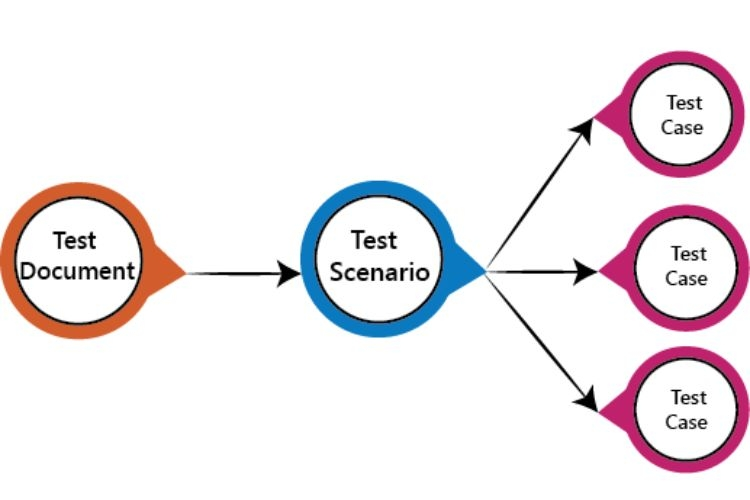
##### **3.1.2.1. Khái niệm**

Kiểm thử tích hợp (Integration test) là quá trình kiểm thử các module của phần mềm thông qua liên kết về logic giữa các module này. Nói một cách đơn giản thì mục tiêu của Integration test là phát hiện ra những vấn đề xoay quanh logic khi những thành phần của phần mềm liên kết lại với nhau. Mỗi module có thể được code bởi nhiều người khác nhau, việc thực hiện Integration test sẽ bao gồm: String testing (kiểm thử chuỗi) và Thread testing (kiểm thử luồng) để kiểm tra sự kết nối mượt mà giữa các thành phần module.

##### **3.1.2.2. Quy trình thực hiện kiểm thử tích hợp**

Trước khi đi vào các bước Integration test, cần đảm bảo các tiêu chí sau:

* Đã tiến hành Unit test với từng module;
* Đã sửa các bug có độ ưu tiên cao;
* Đã có đầy đủ module cần thiết cho phần mềm;
* Các tài liệu kiểm thử tích hợp đã được duyệt;
* Môi trường kiểm thử phù hợp với Integration test;
* Kế hoạch để dự phòng các rủi ro trong quá trình kiểm thử.



Sau đó, các bước cơ bản để tiến hành Integration test như sau:

1. Xem tài liệu Integration test plan và xác định các Module cần test: Bắt đầu bằng việc nghiên cứu tài liệu kế hoạch kiểm thử tích hợp (Integration Test Plan) để hiểu rõ phạm vi và mục tiêu của quá trình kiểm thử. Dựa vào cấu trúc của hệ thống, xác định các module hoặc thành phần cần kiểm tra, cũng như mối quan hệ giữa chúng. Bước này đảm bảo rằng tất cả các phần tích hợp quan trọng đều được kiểm thử đầy đủ.
2. Tạo các kịch bản test (test scenarios), trường hợp test (test cases) và scripts: Xây dựng các kịch bản kiểm thử phản ánh cách các module tương tác với nhau trong các tình huống thực tế. Dựa trên kịch bản, tạo các trường hợp kiểm thử chi tiết với các bước thực hiện, dữ liệu đầu vào, và kết quả mong đợi. Ngoài ra, nếu cần, viết các script tự động để hỗ trợ kiểm thử, đặc biệt với những bài kiểm tra phức tạp.
3. Tiến hành test theo test cases và báo cáo lỗi: Chạy các trường hợp kiểm thử theo kịch bản đã lập, đảm bảo kiểm tra các điểm tích hợp quan trọng giữa các module. Ghi nhận các lỗi phát hiện được trong quá trình kiểm thử, bao gồm thông tin chi tiết như điều kiện xảy ra lỗi, hành động thực hiện, và hậu quả của lỗi.
4. Theo dõi và kiểm thử nhiều lần các trường hợp lỗi để ghi nhận đầy đủ: Sau khi các lỗi được sửa, thực hiện kiểm thử lại các trường hợp liên quan để đảm bảo lỗi đã được khắc phục và không gây ra vấn đề mới. Quá trình này cần được thực hiện cẩn thận để xác minh rằng các module hoạt động ổn định sau mỗi lần sửa lỗi.
5. Lặp lại bước 3 và 4 với các test cases còn lại: Tiếp tục kiểm tra các trường hợp kiểm thử khác theo quy trình đã thực hiện ở các bước trước. Đảm bảo rằng tất cả các kịch bản kiểm thử đều được thực hiện và các lỗi được giải quyết triệt để. Việc này giúp đảm bảo rằng toàn bộ hệ thống hoạt động ổn định và đúng yêu cầu sau khi tích hợp.

Có nhiều loại hoặc cách tiếp cận khác nhau để kiểm thử tích hợp. Các phương pháp phổ biến và được sử dụng thường xuyên nhất là Kiểm thử tích hợp Big Bang, Kiểm thử tích hợp Top-down, Kiểm thử tích hợp từ dưới lên và Kiểm thử tích hợp Bottom up. Sự lựa chọn của phương pháp phụ thuộc vào các yếu tố khác nhau như chi phí, độ phức tạp, mức độ quan trọng của ứng dụng, v.v. Ngoài ra, có nhiều loại thử nghiệm tích hợp ít được biết đến như tích hợp dịch vụ phân tán, thử nghiệm tích hợp sandwich, tích hợp đường trục, tích hợp tần số cao, tích hợp lớp, v.v.

##### **3.1.2.3. Vai trò của kiểm thử tích hợp trong kiểm thử phần mềm**

* Đảm bảo tính thống nhất cho toàn bộ sản phẩm phần mềm: do có thể mỗi module là một Developer khác nhau code, họ có logic và phương pháp thực hiện khác nhau. Nếu không thực hiện Integration test sẽ không có gì đảm bảo rằng những module này được kết hợp hiệu quả mà không báo lỗi;
* Kiểm tra tổng quan về Giao diện và cơ sở dữ liệu của các module khi kết hợp với nhau, nhằm phát hiện sớm các thiếu sót để kịp thời xử lý;
* Kiểm tra độ tương thích đối với hệ thống khi kết hợp các module lại với nhau;
* Nhằm phát hiện thêm những lỗi có thể bị bỏ qua trong quá trình Unit test trước đó.

##### **3.1.2.4. Nhược điểm kiểm thử tích hợp**

* Khó xác định nguyên nhân lỗi: Khi lỗi xảy ra, rất khó để xác định module nào gây ra do các thành phần phụ thuộc lẫn nhau.
* Phức tạp và tốn thời gian: Tích hợp các module, thiết lập môi trường kiểm thử và sửa lỗi phát sinh có thể mất nhiều thời gian.
* Chi phí cao: Yêu cầu tài nguyên lớn như công cụ, nhân lực và môi trường kiểm thử dẫn đến chi phí cao.

#### 3.1.3. Kiểm thử hệ thống

##### **3.1.3.1. Khái niệm**

Kiểm thử hệ thống (System test) là phương pháp quan trọng trong việc theo dõi và đánh giá hoạt động của một sản phẩm hoặc hệ thống phần mềm sau khi nó đã hoàn thiện và tích hợp đầy đủ. Quá trình này dựa trên các đặc tả và yêu cầu chức năng đã được xác định trước để kiểm tra xem hệ thống có hoạt động theo yêu cầu đã đặt ra hay không. Kiểm thử hệ thống thường thực hiện trong hình thức hộp đen, nghĩa là chỉ kiểm tra các tính năng hoạt động bên ngoài của phần mềm. Quá trình kiểm thử này không yêu cầu kiến thức về mã nguồn, lập trình hay thiết kế, mà tập trung hoàn toàn vào cách thức người dùng tương tác với hệ thống.

Các khía cạnh mà System test tập trung vào bao gồm:

* Hiệu suất: Đảm bảo rằng hệ thống phần mềm hoạt động theo yêu cầu mà không gặp lỗi hoặc sự cố.
* Bảo mật: Đảm bảo sự an toàn của sản phẩm trước mọi vi phạm bảo mật, bảo vệ dữ liệu quan trọng của tổ chức.
* Phục hồi: Đảm bảo hệ thống có khả năng phục hồi như mong đợi sau khi có sự cố.
* Giao diện: Kiểm tra đảm bảo rằng giao diện của hệ thống đáp ứng đúng và không gặp sự cố khi các thành phần tích hợp.
* Cài đặt: Đảm bảo sản phẩm có thể được triển khai vào môi trường sản xuất mà không gặp khó khăn.
* Khả dụng: Tối ưu hóa trải nghiệm người dùng là một khía cạnh quan trọng.
* Tài liệu: Kiểm tra độ chính xác của tài liệu liên quan đến sản phẩm.
* Tải/Áp lực: Đảm bảo hệ thống vẫn hoạt động chính xác dưới tải và áp lực khác nhau.

##### **3.1.3.2. Quy trình thực hiện kiểm thử hệ thống**

Dưới đây là những điều kiện tiên quyết quan trọng cho việc thực hiện kiểm thử hệ thống:

* Đảm bảo sự thống nhất trong việc kiểm tra toàn bộ phần mềm.
* Kiểm thử tích hợp trên sản phẩm đã được tiến hành.
* Phần mềm phải được phát triển hoàn thiện.
* Môi trường kiểm thử cần phải chuẩn bị sẵn sàng trước khi bắt đầu quá trình kiểm tra hệ thống.

Mặc dù quá trình kiểm tra hệ thống có thể thay đổi tùy thuộc vào từng dự án, nhưng thường có sáu bước chung được xác định như sau:

1. Tạo Kế hoạch kiểm thử: Đây là bước đầu tiên, trong đó người lãnh đạo hoặc quản lý kiểm thử xác định phạm vi, mục tiêu, chiến lược kiểm tra, quyết định phương pháp kiểm tra (thủ công hoặc tự động), xác định tiêu chí đầu vào và đầu ra, cũng như gán vai trò và trách nhiệm.
2. Tạo Test Case: Tạo các trường hợp thử nghiệm dựa trên các use case và yêu cầu cụ thể như kỹ thuật, giao diện người dùng, chức năng, phi chức năng, hiệu suất, và nhiều yếu tố khác.
3. Lựa chọn Dữ liệu kiểm thử: Khi đã tạo các trường hợp thử, nhóm sẽ lựa chọn hoặc tạo dữ liệu cần thiết để thực hiện kiểm thử. Đây là điều kiện đầu vào quan trọng giúp đạt được kết quả mong muốn.
4. Thực hiện Test Case: Tiến hành kiểm thử các trường hợp thử đã được tạo trước đó, đồng thời ghi nhận bất kỳ sự khác biệt hoặc vấn đề nào phát sinh. Đồng thời, việc ghi nhận kết quả kiểm thử cũng được thực hiện tại đây.
5. Báo cáo và Xử lý lỗi: Báo cáo tất cả các lỗi và sự cố cho các thành viên trong nhóm. Sau khi báo cáo, nhà phát triển hoặc lập trình viên sẽ làm việc cùng nhóm kiểm thử để khắc phục và giải quyết vấn đề.
6. Lặp lại Chu trình Kiểm tra (Nếu cần): Nếu có vấn đề hoặc lỗi, nhóm kiểm thử sẽ lặp lại quá trình kiểm tra để đảm bảo có kết quả như mong đợi.

##### **3.1.3.3. Vai trò của kiểm thử hệ thống trong kiểm thử phần mềm**

* Đảm bảo chất lượng: Kiểm tra hệ thống giúp đảm bảo sản phẩm đáp ứng các tiêu chuẩn chất lượng được đề ra.
* Xác minh yêu cầu: Kiểm tra hệ thống xác minh rằng hệ thống phần mềm đáp ứng các yêu cầu chức năng, kỹ thuật và kinh doanh theo yêu cầu của khách hàng.
* Ngăn ngừa lỗi: Thực hiện kiểm tra từ đầu đến cuối giúp ngăn ngừa lỗi hệ thống và sự cố trong quá trình vận hành trong môi trường thực tế.
* Môi trường tương tự sản phẩm cuối: Kiểm tra trong môi trường tương tự như môi trường sản xuất giúp nhà phát triển và các bên liên quan hiểu rõ hơn về phản ánh của người dùng đối với sản phẩm.
* Cung cấp sản phẩm chất lượng: Chức năng này đóng vai trò quan trọng trong việc cung cấp một sản phẩm chất lượng cho người dùng cuối.
* Kiểm tra yêu cầu và kiến trúc: Trong giai đoạn kiểm thử hệ thống, các yêu cầu nghiệp vụ và kiến trúc ứng dụng được kiểm tra.
* Đầu vào và kết quả như mong đợi: Kiểm tra hệ thống đảm bảo rằng đầu vào được cung cấp sẽ đạt được kết quả theo kỳ vọng.

##### **3.1.3.4. Nhược điểm kiểm thử hệ thống**

* Khó tái tạo lỗi: Lỗi trong môi trường thực tế có thể không dễ tái tạo trong môi trường kiểm thử.
* Tốn thời gian và chi phí: Kiểm tra toàn bộ hệ thống với các tình huống phức tạp đòi hỏi nhiều tài nguyên và thời gian.
* Phạm vi rộng: Việc bao quát hết tất cả các trường hợp sử dụng và yêu cầu có thể không đảm bảo.

#### 3.1.4. Kiểm thử chấp nhận

##### **3.1.4.1. Khái niệm**

Kiểm thử chấp nhận (Acceptance Test) là một phương pháp kiểm thử phần mềm trong đó hệ thống được kiểm thử khả năng chấp nhận. Mục đích chính của kiểm thử này là đánh giá sự tuân thủ của hệ thống với các yêu cầu nghiệp vụ trước khi đưa tới tay người dùng. Đây là một kiểm thử chính thức theo nhu cầu, yêu cầu của người dùng và quy trình được tiến hành để xác định xem hệ thống có đáp ứng các tiêu chí chấp nhận hay không. Và có cho phép người dùng, khách hàng hoặc các đơn vị được ủy quyền khác xác định có chấp nhận hệ thống hay không. Kiểm thử chấp nhận là giai đoạn cuối cùng của kiểm thử phần mềm được thực hiện sau kiểm thử hệ thống và trước khi đưa vào sử dụng thực tế.

Các loại kiểm thử chấp nhận:

* Kiểm thử chấp nhận người dùng (UAT): Kiểm thử này được sử dụng để kiểm thử sự chấp nhận của người dùng xem chúng có hoạt động đáp ứng nhu cầu của họ hay không. Các yêu cầu cụ thể thường được khách hàng sử dụng chủ yếu được chọn cho mục đích kiểm thử. Đây cũng được gọi là kiểm thử người dùng cuối.
* Kiểm thử chấp nhận kinh doanh (BAT): Được sử dụng để xác định xem sản phẩm có đáp ứng mục đích kinh doanh hay không. BAT chủ yếu tập trung vào lợi nhuận kinh doanh vốn khác thách thức do điều kiện thị trường thay đổi và công nghệ mới nên việc triển khai hiện tại có thể phải thay đổi dẫn đến tăng ngân sách.
* Kiểm thử chấp nhận hợp đồng (CAT): CAT là một hợp đồng quy định rằng một khi sản phẩm đi vào hoạt động, trong một khoảng thời gian xác định trước, kiểm thử chấp nhận phải được thực hiện và nó phải vượt qua tất cả các trường hợp sử dụng chấp nhận.
* Kiểm thử chấp nhận quy định (RAT): Được sử dụng để xác định xem sản phẩm có vi phạm các quy tắc và quy định được xác định bởi chính phủ quốc gia nơi sản phẩm được phát hành hay không. Điều này có thể vô tình nhưng có thể sẽ tác động tiêu cực đến doanh nghiệp.
* Kiểm thử chấp nhận hoạt động (OAT): Được sử dụng để xác định mức độ sẵn sàng hoạt động của sản phẩm và là kiểm thử phi chức năng. Nó chủ yếu bao gồm kiểm thử khả năng phục hồi, khả năng tương thích, khả năng bảo trì, độ tin cậy. OAT đảm bảo tính ổn định của sản phẩm trước khi nó được đưa vào sản xuất.
* Kiểm thử alpha: Được sử dụng để xác định sản phẩm trong môi trường kiểm thử phát triển bởi một nhóm Tester chuyên biệt thường được gọi là người kiểm thử alpha.
* Kiểm thử beta: Được sử dụng để đánh giá sản phẩm bằng cách hiển thị sản phẩm cho người dùng cuối thực sự, thường được gọi là người thử nghiệm beta trong môi trường của họ. Phản hồi được thu thập từ người dùng và các lỗi được sửa. Ngoài ra, điều này giúp nâng cao sản phẩm để mang lại trải nghiệm người dùng phong phú.

##### **3.1.4.2. Quy trình thực hiện kiểm thử chấp nhận**

Trước khi đi vào các bước kiểm thử chấp nhận, cần đảm bảo các tiêu chí sau:

* Phải đảm bảo các yêu cầu nghiệp vụ chính của ứng dụng hoạt động
* Phần mềm đã được hoàn thiện nhất
* Các khâu kiểm thử Unit testing, integration testing, system testing đã được hoàn thành
* Không có lỗi quan trọng còn tồn tại trong hệ thống
* Lỗi về thẩm mỹ được chấp nhận trước UAT
* Regression testing phải được hoàn thành và không có lỗi lớn
* Tất cả các lỗi đã phát hiện phải được sửa và kiểm tra trước khi UAT
* Môi trường Acceptance Testing phải được chuẩn bị sẵn sàng
* Nhà phát triển phải chắc chắn rằng hệ thống đã sẵn sàng thực hiện Acceptance Testing

Sau đó, các bước cơ bản để tiến hành kiểm thử chấp nhận như sau:

1. Phân tích các yêu cầu nghiệp vụ của phần mềm: Bước đầu tiên là đọc và hiểu rõ các yêu cầu nghiệp vụ từ tài liệu hoặc từ khách hàng. Đây là giai đoạn quan trọng để xác định các chức năng mà phần mềm phải đáp ứng, cũng như các tiêu chí chấp nhận mà hệ thống cần đạt được. Việc phân tích kỹ lưỡng giúp tránh hiểu sai yêu cầu, đồng thời làm cơ sở để xây dựng các kịch bản kiểm thử chính xác.
2. Tạo kế hoạch kiểm tra Acceptance Testing: Dựa trên yêu cầu đã phân tích, lập kế hoạch kiểm thử bao gồm phạm vi kiểm thử, mục tiêu, và các tiêu chí chấp nhận. Kế hoạch cần chỉ rõ thời gian, công cụ, và nhân lực cần thiết cho quá trình kiểm thử. Ngoài ra, xác định rõ trách nhiệm của từng thành viên trong nhóm giúp đảm bảo mọi người hiểu rõ vai trò của mình trong quá trình kiểm thử.
3. Xác định các kịch bản kiểm thử: Kịch bản kiểm thử mô tả các tình huống thực tế mà người dùng cuối sẽ thực hiện với phần mềm. Các kịch bản này phải phản ánh các trường hợp sử dụng quan trọng nhất, đặc biệt là những trường hợp liên quan đến yêu cầu nghiệp vụ chính. Việc xác định kịch bản kỹ càng giúp quá trình kiểm thử có trọng tâm và hiệu quả hơn.
4. Tạo các trường hợp kiểm tra Acceptance Testing: Trường hợp kiểm thử là các bước kiểm tra cụ thể dựa trên từng kịch bản kiểm thử. Mỗi trường hợp cần xác định rõ dữ liệu đầu vào, hành động thực hiện, và kết quả mong đợi. Đảm bảo các trường hợp kiểm thử bao gồm cả các tình huống thành công và thất bại để kiểm tra đầy đủ chức năng của hệ thống.
5. Chuẩn bị data test (giống với data thật nhất): Dữ liệu kiểm thử phải sát với dữ liệu thực tế mà hệ thống sẽ sử dụng khi vận hành. Điều này giúp đảm bảo các tình huống kiểm thử phản ánh đúng điều kiện thực tế, từ đó phát hiện sớm các lỗi hoặc hạn chế trong hệ thống. Việc chuẩn bị dữ liệu cũng cần tuân thủ các quy định về bảo mật, nhất là khi sử dụng dữ liệu nhạy cảm.
6. Thực hiện kiểm thử: Đây là giai đoạn chạy các trường hợp kiểm thử trên hệ thống. Thực hiện kiểm thử phải tuân theo kế hoạch đã xây dựng, đồng thời ghi nhận các lỗi hoặc sự khác biệt so với kết quả mong đợi. Quá trình này có thể yêu cầu lặp đi lặp lại để đảm bảo hệ thống hoạt động đúng như yêu cầu.
7. Ghi nhận kết quả: Tất cả các kết quả kiểm thử, bao gồm các trường hợp thành công và thất bại, cần được ghi nhận cẩn thận. Ghi nhận bao gồm thông tin chi tiết về lỗi (nếu có), điều kiện xảy ra lỗi, và các bước để tái tạo lỗi. Điều này giúp việc sửa lỗi và kiểm tra lại trở nên dễ dàng hơn.
8. Báo cáo kết quả: Báo cáo kiểm thử tổng hợp tất cả các thông tin từ quá trình kiểm thử, bao gồm các trường hợp đã kiểm tra, số lượng lỗi phát hiện, và trạng thái hiện tại của hệ thống. Báo cáo này là tài liệu quan trọng để quyết định liệu phần mềm có đáp ứng yêu cầu và sẵn sàng phát hành hay không.

##### **3.1.4.3. Vai trò của kiểm thử chấp nhận trong kiểm thử phần mềm**

* Kiểm thử chấp nhận giúp xác định và kiểm chứng yêu cầu người dùng: Đây là giai đoạn để xác minh trực tiếp rằng hệ thống đáp ứng đúng các yêu cầu và kỳ vọng ban đầu của người dùng. Thông qua kiểm thử này, các nhóm phát triển và khách hàng có thể đảm bảo rằng sản phẩm cuối cùng phù hợp với mục đích sử dụng thực tế.
* Phát hiện các vấn đề tiềm ẩn mà Unit Test hoặc Integration Test chưa nhận ra: Acceptance Testing giúp nhận diện những lỗi hoặc thiếu sót mà các giai đoạn kiểm thử trước đó có thể đã bỏ sót, đặc biệt là các vấn đề liên quan đến trải nghiệm và hoạt động tổng thể của hệ thống trong môi trường thực tế.
* Cung cấp cái nhìn tổng quan về hệ thống: Người dùng và các bên liên quan có cơ hội đánh giá toàn bộ hệ thống qua góc nhìn thực tiễn. Kiểm thử chấp nhận không chỉ kiểm tra chức năng mà còn xác định mức độ đáp ứng của hệ thống đối với các yêu cầu vận hành và hiệu năng.
* Xác minh và xác nhận nhu cầu của khách hàng: Đây là bước cuối cùng để đảm bảo rằng hệ thống thực sự đáp ứng đầy đủ nhu cầu của khách hàng. Acceptance Testing là cơ sở quan trọng để quyết định hệ thống đã sẵn sàng triển khai hay cần cải tiến thêm.

##### **3.1.4.4. Nhược điểm kiểm thử chấp nhận**

* Người dùng nên có kiến ​​thức cơ bản về sản phẩm hoặc ứng dụng.
* Đôi khi, người dùng không muốn tham gia vào quá trình kiểm thử.
* Phản hồi cho kiểm thử mất nhiều thời gian vì nó liên quan đến nhiều người dùng và ý kiến ​​có thể khác nhau giữa người dùng này với người dùng khác.
* Nhóm phát triển không tham gia vào quá trình kiểm thử này.

### 3.2. Các phương pháp kiểm thử

#### 3.2.1. Kiểm thử hộp trắng (White box testing)

##### **3.2.1.1. Khái niệm**

Kiểm thử hộp trắng (White box testing) là phương pháp kiểm thử phần mềm nhằm kiểm tra thuật toán, cấu trúc code bên trong của sản phẩm. Hộp trắng tượng trưng cho khả năng nhìn xuyên qua lớp vỏ bên ngoài của phần mềm để thấy được hoạt động bên trong của chúng.

Mục đích của White box testing là đảm bảo tất cả các câu lệnh và điều kiện sẽ được thực hiện đúng, kết quả đầu ra như mong đợi. Tester sẽ xác minh luồng hoạt động cho ứng dụng bằng cách kiểm tra một loạt đầu vào (Input) đã được xác định từ trước có dẫn đến đầu ra (Output) như dự kiến không? Nếu Output không khớp với kỳ vọng, có nghĩa là sản phẩm đang bị lỗi.

Am hiểu về lập trình, có kiến thức về công nghệ thông tin là điều kiện tiên quyết để tester có thể tiến hành kiểm thử hộp trắng. Thông thường, nhiệm vụ thực thi White box Testing sẽ do chính các Developers đảm nhiệm.

##### **3.2.1.2. Các kỹ thuật kiểm thử hộp trắng phổ biến**

* Kiểm thử đường cơ bản - Đồ thị dòng (Basic Path Testing - Control Flow Graph): Kỹ thuật này sử dụng đồ thị dòng (control flow graph) để mô tả tất cả các dòng đi (path) có thể có trong chương trình. Mỗi đường trong đồ thị này đại diện cho một chuỗi các câu lệnh được thực thi từ đầu đến cuối. Mục tiêu của kiểm thử đường cơ bản là đảm bảo rằng mỗi đường đi trong đồ thị dòng được kiểm tra ít nhất một lần, từ đó giúp phát hiện lỗi trong các cấu trúc điều khiển (như vòng lặp, câu lệnh điều kiện). Kiểm thử này tập trung vào việc kiểm tra độ phủ của các đường đi cơ bản trong chương trình.
* Kiểm thử dựa trên luồng điều khiển (Control Flow Testing): Kiểm thử dựa trên luồng điều khiển nhằm kiểm tra các dòng điều khiển trong mã nguồn, chẳng hạn như các câu lệnh điều kiện (if, else), các vòng lặp, và các nhánh (branching). Kỹ thuật này giúp xác định các lỗi logic trong cách chương trình xử lý các điều kiện và các nhánh. Bằng cách kiểm tra các nhánh khác nhau của chương trình, kiểm thử này giúp phát hiện các lỗi liên quan đến việc chương trình không xử lý đúng các tình huống khác nhau hoặc không đáp ứng các điều kiện cụ thể.
* Kiểm thử dựa trên luồng dữ liệu (Data-Flow Testing): Kiểm thử luồng dữ liệu tập trung vào cách dữ liệu di chuyển và được xử lý trong chương trình. Điều này bao gồm việc kiểm tra các biến, cách chúng được khởi tạo, sử dụng, thay đổi và loại bỏ trong quá trình thực thi chương trình. Kỹ thuật này nhằm phát hiện lỗi liên quan đến việc sử dụng hoặc thay đổi dữ liệu không đúng cách, như lỗi không khởi tạo biến, hoặc việc sử dụng dữ liệu không hợp lệ. Kiểm thử này cũng giúp đảm bảo rằng tất cả các luồng dữ liệu trong chương trình được xử lý một cách hợp lý và không gây ra lỗi.
* Kiểm thử đột biến (Mutation Testing): Kiểm thử đột biến là một kỹ thuật trong đó mã nguồn của phần mềm được "đột biến" (biến đổi một cách có chủ đích) để tạo ra các phiên bản lỗi của chương trình. Mục tiêu của kỹ thuật này là kiểm tra khả năng phát hiện lỗi của bộ kiểm thử. Các biến thể lỗi này, gọi là "mutants", sẽ thay đổi một phần mã nguồn như thay đổi toán tử, điều kiện hoặc giá trị, và sau đó bộ kiểm thử sẽ kiểm tra xem liệu các mutants này có bị phát hiện không. Kiểm thử đột biến giúp đánh giá chất lượng của bộ kiểm thử và xác minh rằng các trường hợp kiểm thử đủ mạnh để phát hiện lỗi trong chương trình.

##### **3.2.1.3. Ưu điểm của kiểm thử hộp trắng**

* Phát hiện lỗi sớm: Kiểm thử hộp trắng cho phép phát hiện lỗi và sai sót ngay từ giai đoạn đầu của quá trình phát triển. Việc phát hiện sớm này giúp tiết kiệm thời gian, công sức và nguồn lực, vì việc sửa lỗi ở giai đoạn sau thường phức tạp và tốn kém hơn.
* Tối ưu hóa mã nguồn: Giúp nhận diện các đoạn mã dư thừa và các khu vực trong phần mềm có thể được tối ưu hóa. Điều này góp phần tạo ra phần mềm hiệu quả hơn và được tổ chức hợp lý hơn.
* Bảo mật: Phát hiện các lỗ hổng bảo mật trong mã nguồn. Bằng cách kiểm tra cấu trúc bên trong của phần mềm, người kiểm thử có thể nhận diện các rủi ro bảo mật tiềm ẩn và đảm bảo các nguyên tắc bảo mật đã được tuân thủ.
* Kiểm tra toàn diện: Kiểm thử hộp trắng xem xét toàn bộ hoạt động nội bộ của phần mềm. Sự toàn diện này đảm bảo mọi phần của mã nguồn đều được kiểm tra và xác thực, giúp phần mềm trở nên mạnh mẽ và đáng tin cậy.
* Đảm bảo chất lượng: Đây là một phần quan trọng trong việc đảm bảo chất lượng phần mềm. Bằng cách kiểm tra cấu trúc nội bộ, kiểm thử hộp trắng đảm bảo rằng phần mềm hoạt động như mong đợi và đáp ứng các tiêu chuẩn đã đặt ra.

##### **3.2.1.4. Nhược điểm của kiểm thử hộp trắng**

* Đòi hỏi nguồn lực chuyên nghiệp có kiến thức và tay nghề cao về lập trình.
* Tốn khá nhiều thời gian để kiểm tra được chi tiết cấu trúc, thuật toán bên trong sản phẩm. Các ứng dụng có cấu trúc hệ thống càng lớn càng cần rất nhiều thời gian để kiểm tra đầy đủ.
* Kiểm thử diễn ra khi sản phẩm chưa được hoàn thiện nên các công cụ phục vụ cho mọi loại triển khai/nền tảng thường không sẵn có.
* Kiểm tra hộp trắng được đánh giá khá phức tạp và tốn kém. Developers không kiểm tra hộp trắng chi tiết có thể dẫn đến lỗi sản xuất.

#### 3.2.2. Kiểm thử hộp đen (Black box testing)

##### **3.2.2.1. Khái niệm**

Kiểm thử hộp đen (Black Box Testing) là phương pháp kiểm thử phần mềm tập trung vào kiểm tra các chức năng của phần mềm mà không cần biết cấu trúc bên trong của mã nguồn. Trong quá trình kiểm thử này, người kiểm thử tập trung vào đầu vào và đầu ra dự kiến mà không quan tâm đến cách phần mềm xử lý dữ liệu hoặc logic nội bộ.

Các chuyên gia kiểm thử hộp đen tập trung vào xác định các lỗi chức năng, sai sót trong giao diện người dùng, tương tác với hệ thống và các lỗi phản hồi từ phần mềm. Họ thiết lập các ca kiểm thử dựa trên yêu cầu, tài liệu mô tả và các quy định, sau đó thực hiện kiểm thử dựa trên dữ liệu đầu vào và đầu ra kỳ vọng.

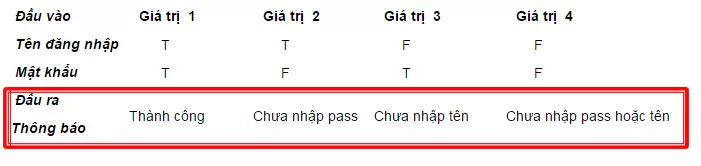
Mục tiêu của kiểm thử hộp đen là đảm bảo rằng phần mềm hoạt động chính xác theo yêu cầu mà không cần biết chi tiết cấu trúc nội bộ của phần mềm. Điều này giúp đảm bảo tính đầy đủ và hiệu quả của kiểm thử, đồng thời phản ánh góc nhìn của người dùng cuối về sản phẩm phần mềm.

Kiểm thử hộp đen có nhiều loại khác nhau, cụ thể là:

* Functional Testing: Kiểm thử chức năng tập trung vào việc kiểm tra xem phần mềm hoạt động chính xác theo yêu cầu chức năng hay không. Các ca kiểm thử chức năng sẽ kiểm tra thành công, thất bại và các điều kiện giới hạn.
* Regression Testing: Kiểm thử hồi quy tập trung vào việc kiểm tra xem các thay đổi trong mã nguồn và môi trường có ảnh hưởng đến các chức năng hiện có hay không
* Usability Testing: Kiểm thử tính sử dụng tập trung vào đánh giá trải nghiệm người dùng khi tương tác với phần mềm, đảm bảo giao diện và trải nghiệm người dùng được tối ưu.
* Stress Testing: Kiểm thử tải tập trung vào đánh giá hệ thống và hiệu suất của phần mềm dưới điều kiện tải và áp lực cao.
* Boundary Testing: Kiểm thử ranh giới tập trung vào kiểm tra lỗi xảy ra khi dữ liệu đầu vào ở biên giới hoặc gần biên giới của các dải giá trị hợp lệ.
* Ad-hoc Testing: Kiểm thử tạm thời tập trung vào việc kiểm tra một cách không có kế hoạch, thường được thực hiện khi cần kiểm tra nhanh chóng hoặc trong trường hợp đặc biệt.
* Compatibility Testing: Kiểm thử tương thích tập trung vào việc kiểm tra phần mềm hoạt động đúng đắn trên các nền tảng phần cứng và phần mềm khác nhau.

##### **3.2.2.2. Các kỹ thuật kiểm thử hộp đen phổ biến**

* Phân vùng tương đương (Equivalence Partitioning): Đây là một kỹ thuật thiết kế kiểm thử phần mềm bao gồm việc chia các giá trị đầu vào thành các phân vùng hợp lệ và không hợp lệ và chọn các giá trị đại diện từ mỗi phân vùng làm dữ liệu kiểm tra.
* Mục đích : Giảm đáng kể số lượng test case cần phải thiết kế vì với mỗi lớp tương đương ta chỉ cần test trên các phần tử đại diện.
* Thiết kế Test-case bằng phân lớp tương đương tiến hành theo 2 bước: (1). Xác định các lớp tương đương (2). Xác định các ca kiểm thử
* Nguyên tắc: 1 lớp các giá trị lớn hơn 1 lớp các giá trị nhỏ hơn n lớp các giá trị hợp lệ
* Ví dụ minh họa: Thiết kế test case sao cho khi người dùng nhập user vào ô text thì chỉ cho nhập số ký tự [6 – 20 ]. Đáp án: Do yêu cầu của bài toán chỉ cho phép nhập số ký tự vào trong khi nhập của user nằm [6 - 20] nên ta có tình huống kiểm thử sau: Nhập vào một trường hợp hợp lệ:nhập 7 ký tự. Nhập vào trường hợp không hợp lệ thứ nhất: nhập 5 ký tự. Nhập vào trường hợp không hợp lệ thứ hai: nhập vào 21 ký tự. Trường hợp đặc biệt: không nhập gì vào ô text đó (để trống).
* Phân tích giá trị biên: Đây là phương pháp test mà chúng ta sẽ test tất cả các giá trị ở vùng biên của dữ liệu vào và dữ liệu ra. Chúng ta sẽ tập trung vào các giá trị biên chứkhông test toàn bộ dữ liệu. Thay vì chọn nhiều giá trị trong lớp đương tươngđể làm đại diện, phân tích giá trị biên yêu cầu chọn một hoặc vài giá trị là các cạnh của lớp tương đương để làm điều kiện test. Phân tích giá trị biên là kỹ thuật thiết kế test case và hoàn thành phân vùng tương đương. Mục tiêu là lựa chọn các test case để thực thi giá trị biên. Phân tích giá trị biên sẽ chọn các giá trị:
* Giá trị nhỏ nhất
* Giá trị ngay trên giá trị nhỏ nhất
* Giá trị bình thường
* Giá trị ngay dưới giá trị lớn nhất
* Giá trị lớn nhất Ví dụ: Cho một mảng [ -3 , 10] ta có thể thiết kế được các test case là: Đáp án: Giá trị nhỏ nhất: -3 Giá trị lớn nhất: 10 Giá trị nhỏ hơn giá trị nhỏ nhất: -4 Giá trị lớn hơn giá trị lớn nhất: 11 Giá trị nằm trong -3 và 10: 0
* Sử dụng bảng quyết định: Một điểm yếu của hai phương pháp trên là chúng không khảo sát sự kết hợp của các trường hợp đầu vào. Việc kiểm tra sự kết hợp đầu vào không phải là một nhiệm vụ đơn giản bởi vì nếu bạn phân lớp tương đương các trạng thái đầu vào thì số lượng sự kết hợp thường là rất lớn. Bảng quyết định sử dụng mô hình các quan hệ logic giữa nguyên nhân và kết quả cho thành phần phần mềm. Mỗi nguyên nhân được biểu diễn như một điềukiện (đúng hoặc sai) của một đầu vào, hoặc kết hợp các đầu vào. Mỗi kết quả được biểu diễn như là một biểu thức Bool biểu diễn một kết quả tương ứng cho những thành phần vừa thực hiện. Kỹ thuật gồm có 4 bước:
* Xác định điều kiện vào và hành động cho mỗi module cần kiểm định.
* Xác định đồ thị nguyên nhân – kết quả.
* Đồ thị được chuyển thành bảng quyết định.
* Những phần trong bảng quyết định được chuyển thành test case. Ví dụ: Trên màn hình đăng nhập, có 2 thông tin cần đưa vào là Tên đăng nhập và mật khẩu, chỉ thực hiện đăng nhập thành công nếu nhập đúng cả Tên đăng nhập và mật khẩu. Các trường hợp còn lại đăng nhập không thành công: Đáp án:



* Kiểm
* Đoán lỗi: Trong kiểm thử phần mềm, đoán lỗi - error guessing - là một phương pháp kiểm thử, trong đó các trường hợp kiểm thử - test case - được sử dụng để tìm lỗi trong các chương trình đã được phát triển - đã code - dựa vào kinh nghiệm trong các lần kiểm thử trước. Phạm vi của các trường hợp kiểm thử thường được dựa vào các kiểm thử viên - tester - có kiến thức liên quan, là những người đã có kinh nghiệm sử dụng và trực giác để xác định những tình huống thường gây ra lỗi trong phần mềm. Các lỗi điển hình như chia cho không, null pointer, hoặc các biến không hợp lệ. Đoán Lỗi không có quy tắc rõ ràng để kiểm thử, test case có thể được thiết kế tùy thuộc vào tình hình, hoặc hoặc luồng công việc trong các tài liệu mô tả chức năng hoặc khi một lỗi không mong muốn / không được mô tả trong tài liệu được tìm thấy trong khi hoạt động kiểm thử. Đoán Lỗi không có quy tắc, nó chỉ sử dụng các kỹ năng kiểm thử trước đó. Trong kiểm thử phần mềm, đoán lỗi có thể nghĩ đến các tình huống nơi mà phần mềm sẽ thất bại. Ví dụ: Chia cho không Nhấn nút gửi trên mẫu đơn mà không cần điền vào bất kỳ mục. Nhập các dữ liệu đặc biệt vào các ô nhập liệu và sau đó kiểm tra hành vi của phần mềm.

##### **3.2.2.3. Ưu điểm của kiểm thử hộp đen**

* Phù hợp và hiệu quả khi số lượng các dòng lệnh của hệ thống là lớn: Kiểm thử hộp đen rất hữu ích trong việc kiểm thử các hệ thống phức tạp và có số lượng dòng lệnh lớn, vì kỹ thuật này tập trung vào đầu vào và đầu ra của hệ thống mà không cần quan tâm đến mã nguồn. Điều này giúp đảm bảo rằng các chức năng hệ thống hoạt động đúng mà không cần phải kiểm tra chi tiết mã nguồn.
* Không cần truy cập mã nguồn: Kiểm thử hộp đen không yêu cầu tester phải hiểu mã nguồn hoặc có kỹ năng lập trình cao, điều này làm cho nó trở nên dễ dàng hơn cho những người không phải là lập trình viên, chẳng hạn như người kiểm thử chức năng hoặc người dùng cuối.
* Phân biệt rõ ràng quan điểm của người dùng với quan điểm của nhà phát triển thông qua các vai trò được xác định rõ ràng: Kiểm thử hộp đen giúp phân biệt rõ ràng giữa cách người dùng tương tác với hệ thống và cách mà nhà phát triển triển khai hệ thống. Điều này giúp người kiểm thử hiểu và kiểm tra phần mềm từ góc độ người dùng, do đó phản ánh đúng cách phần mềm sẽ được sử dụng trong thực tế.
* Một số lượng lớn tester có kỹ năng vừa phải có thể kiểm tra ứng dụng mà không cần có nhiều kiến thức, ngôn ngữ lập trình hoặc hệ điều hành: Kiểm thử hộp đen không yêu cầu kiến thức chuyên sâu về ngôn ngữ lập trình hoặc hệ điều hành. Điều này cho phép tuyển dụng một lượng lớn tester với kỹ năng vừa phải, giúp tăng khả năng kiểm thử và giảm chi phí đào tạo.

##### **3.2.2.4. Nhược điểm của kiểm thử hộp đen**

* Yêu cầu dữ liệu đầu vào với một khối lượng mẫu lớn: Kiểm thử hộp đen đòi hỏi một lượng lớn dữ liệu đầu vào để đảm bảo tính toàn diện của quá trình kiểm thử. Việc chuẩn bị và xử lý một số lượng lớn mẫu đầu vào có thể trở nên phức tạp và tốn thời gian, đặc biệt đối với các hệ thống có tính năng phong phú.
* Khó khăn trong việc viết kịch bản kiểm thử: Việc tạo ra các kịch bản kiểm thử cho kiểm thử hộp đen đòi hỏi phải xác định toàn bộ các yếu tố đầu vào có thể xảy ra. Điều này không chỉ mất nhiều thời gian mà còn rất dễ gây nhầm lẫn, nhất là khi không có đủ thông tin chi tiết về cách hệ thống xử lý các đầu vào này.
* Khả năng lạc lối của kỹ sư trong quá trình kiểm thử: Vì kiểm thử hộp đen chủ yếu tập trung vào việc kiểm tra đầu vào và đầu ra mà không nhìn vào mã nguồn, kỹ sư kiểm thử có thể dễ dàng bị lạc lối trong quá trình kiểm thử. Điều này có thể dẫn đến việc bỏ sót các tình huống quan trọng hoặc không phát hiện được lỗi tiềm ẩn trong các phần mã nguồn phức tạp.

#### 3.2.3. Kiểm thử hộp xám (Gray box testing)

##### **3.2.3.1. Khái niệm**

Kiểm thử hộp xám là phương pháp kết hợp giữa kiểm thử hộp đen và kiểm thử hộp trắng. Muốn kiểm thử hộp xám, các Tester phải có kiến thức cơ bản về các luồng hoạt động bên trong hệ thống. Khi kiểm tra hộp xám, Tester có thể truy cập vào các tài liệu thiết kế hệ thống và cơ sở dữ liệu của hệ thống. Trong kiểm thử hộp xám, Tester chỉ nhìn thấy được một phần bên trong hệ thống. Vì hộp xám trong kiểm thử được thiết kế kiểu bán trong suốt, nhìn từ ngoài vào chỉ thấy một phần bên trong. Phương pháp kiểm thử hộp xám được dùng chủ yếu trong kiểm thử tích hợp.

##### **3.2.3.2. Các kỹ thuật kiểm thử hộp xám phổ biến**

* Kiểm thử ma trận (Matrix Testing): Kiểm thử ma trận là một kỹ thuật kiểm thử giúp xác định tất cả các biến có sẵn trong hệ thống và đảm bảo rằng mọi trường hợp đầu vào đều được kiểm tra. Bằng cách ghi lại ma trận với đầy đủ các biến và dữ liệu đầu vào, kiểm thử này giúp đảm bảo rằng không có trường hợp nào bị bỏ sót. Ma trận kiểm thử sẽ là danh sách các trường hợp kiểm thử, giúp tổ chức và theo dõi quá trình kiểm tra một cách rõ ràng.
* Kiểm thử hồi quy (Regression Testing): Kiểm thử hồi quy được thực hiện khi có sự thay đổi hoặc bổ sung chức năng mới trong hệ thống. Mục tiêu là đảm bảo rằng chức năng mới không làm ảnh hưởng đến các chức năng đã có. Quá trình kiểm thử hồi quy thường bao gồm việc kiểm tra lại các trường hợp kiểm thử trước đây để xác định xem sự thay đổi có làm hỏng chức năng cũ hay không. Đây là một phần quan trọng trong việc duy trì chất lượng phần mềm trong suốt quá trình phát triển.
* Kiểm thử mảng trực giao (Orthogonal Array Testing or OAT): Kiểm thử mảng trực giao là một kỹ thuật thống kê giúp tạo ra các hoán vị của các đầu vào để tối ưu hóa việc kiểm thử. Mục tiêu là giảm thiểu công sức trong giai đoạn lập kế hoạch và thiết kế kiểm thử bằng cách tạo ra các test case với phạm vi kiểm tra tối ưu. Kỹ thuật này giúp tiết kiệm thời gian và tài nguyên khi kiểm thử các kết hợp đầu vào phức tạp, vì nó chỉ chọn các hoán vị quan trọng nhất, giảm số lượng trường hợp kiểm thử cần thiết.
* Kiểm thử mẫu (Pattern Testing): Kiểm thử mẫu được thực hiện bằng cách phân tích các mẫu lỗi từ dữ liệu lịch sử hệ thống trước đó. Không giống như kiểm thử hộp đen, kiểm thử hộp xám đi sâu vào mã nguồn và tìm ra nguyên nhân gốc rễ của các sự cố đã xảy ra. Bằng cách phân tích các lỗi trong quá khứ, kỹ thuật này giúp nhận diện các mô hình lỗi và đưa ra giải pháp ngăn ngừa chúng trong tương lai, đồng thời cải thiện chất lượng phần mềm.

##### **3.2.3.3. Ưu điểm của kiểm thử hộp xám**

* Quan điểm kiểm thử từ phía người dùng: Kiểm thử hộp xám tiếp cận phần mềm từ góc độ người dùng, giúp xác định liệu phần mềm có thực sự đáp ứng được yêu cầu và mong muốn của người dùng cuối hay không. Điều này đảm bảo rằng phần mềm hoạt động đúng như kỳ vọng của người sử dụng.
* Kết hợp lợi ích của cả hộp đen và hộp trắng: Kiểm thử hộp xám kết hợp những lợi ích của cả hai phương pháp kiểm thử hộp đen (kiểm thử từ ngoài vào, không cần mã nguồn) và hộp trắng (kiểm thử từ trong ra, có thể truy cập mã nguồn). Điều này giúp đảm bảo tính toàn diện và hiệu quả hơn trong việc phát hiện lỗi.
* Dựa trên các đặc tả chức năng và sơ đồ kiến trúc hệ thống: Kiểm thử hộp xám sẽ dựa trên các đặc tả chức năng và mô tả người dùng, cũng như sơ đồ kiến trúc hệ thống, từ đó xác nhận các yêu cầu ngay từ giai đoạn đầu. Điều này giúp phát hiện lỗi ngay khi hệ thống chưa hoàn thiện.
* Tính minh bạch trong kiểm tra: Việc kiểm tra sẽ trở nên tường minh vì có sự hợp tác giữa người kiểm thử và kỹ sư thiết kế hoặc phát triển, giúp nâng cao tính chính xác và hiệu quả của quá trình kiểm thử.

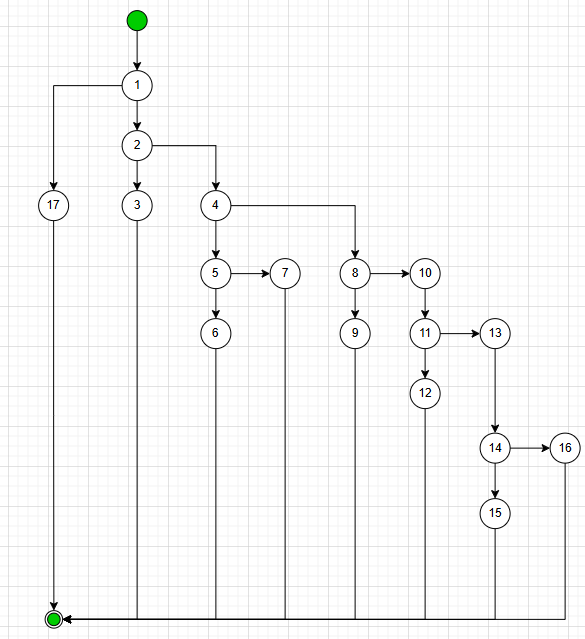
##### **3.2.3.4. Nhược điểm của kiểm thử hộp xám**

* Khó khăn trong việc liên kết lỗi trong hệ thống phân tán: Kiểm thử hộp xám có thể gặp khó khăn trong việc xác định và liên kết lỗi khi thực hiện kiểm thử trên các ứng dụng có hệ thống phân tán. Điều này có thể làm cho quá trình kiểm thử trở nên phức tạp và khó khăn hơn.
* Phạm vi kiểm thử thấp hơn so với kiểm thử hộp đen và hộp trắng riêng biệt: Kiểm thử hộp xám thường dẫn đến phạm vi kiểm thử thấp hơn so với khi thực hiện kiểm thử hộp đen và hộp trắng riêng biệt. Điều này có thể làm giảm khả năng phát hiện các lỗi tiềm ẩn trong hệ thống.
* Không phù hợp với một số loại chức năng: Kiểm thử hộp xám có thể không phù hợp với một số loại chức năng hoặc tình huống, đặc biệt là những chức năng yêu cầu kiểm thử chi tiết và chuyên sâu vào mã nguồn, điều này đòi hỏi sử dụng phương pháp hộp trắng.

## **CHƯƠNG 4: KIỂM THỬ HỘP TRẮNG**

### 4.1. Hàm checkSeeCandiate.

|  |
| --- |
| let checkSeeCandiate = (data) => {  return new Promise(async (resolve, reject) => {  try (1) {  if (!data.userId && !data.companyId) (2) {  resolve({  errCode: 1,  errMessage: 'Missing required parameters !'  }) (3)  } else (4) {  let company  if (data.userId !== 'null') (5) {  let user = await db.User.findOne({  where: {id: data.userId},  attributes: {  exclude: ['userId']  }  })  company = await db.Company.findOne({  where : {id: user.companyId},  attributes: ['id','allowCV','allowCvFree'],  raw: false  }) (6)  }  else (7) {  company = await db.Company.findOne({  where: { id: data.companyId },  attributes: ['id','allowCV','allowCvFree'],  raw: false  })  }  if (!company) (8) {  resolve({  errCode: 2,  errMessage: "Không tìm thấy công ty người dùng sở hữu"  }) (9)  }  else (10) {  if (company.allowCvFree > 0) (11) {  company.allowCvFree -= 1  await company.save()  resolve({  errCode: 0,  errMessage: "Ok"  }) (12)  }  Else (13) if (company.allowCV > 0) (14) {  company.allowCV -= 1  await company.save()  resolve({  errCode: 0,  errMessage: "Ok"  }) (15)  }  Else (16) {  resolve({  errCode: 1,  errMessage: "Công ty bạn đã hết lượt xem"  })  }    }  }  } catch (17) (error) {  reject(error)  }  })  } |

* + 1. **Lập đồ thị dòng điều khiển cơ bản.**
    2. **Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 7 nút quyết định nhị phân nên có độ phức tạp C = 7+1 = 8.

Vậy có ít nhất là 8 test case có thể bao phủ 100% các nhánh

* + 1. **Thiết kế test cases cho từng đường.**

Test case cho đường 1: 1->17

Value(u,c): undefined, undefined

Kết quả kỳ vọng: in ra màn hình 'Missing required parameters !'

Test case cho đường 2: 1->2->3

Value(u,c): null, null

Kết quả kỳ vọng: exclude: 'Missing required parameters !'

Test case cho đường 3: 1->2->4->5->6

Value(u,c): 123,null

Kết quả kỳ vọng: Xử lý thành công, tìm thấy thông tin công ty qua userId

Test case cho đường 4: 1->2->4->5->7

Value(u,c): null,456

Kết quả kỳ vọng: Xử lý thành công, tìm thấy thông tin công ty qua companyId.

Test case cho đường 5: 1->2->4->8->9

Value(u,c): 999,null

Kết quả kỳ vọng: In ra màn hình 'Không tìm thấy công ty người dùng sở hữu'.

Test case cho đường 6: 1->2->4->8->10->11->12

Value(u,c,"allowCvFree", "allowCV"):123,null,1,0

Kết quả kỳ vọng: In ra màn hình 'Ok' và giảm giá trị allowCvFree xuống 1.

Test case cho đường 7: 1->2->4->8->10->11->13->14->15

Value(u,c, ,"allowCvFree", "allowCV"):123,null,0,1

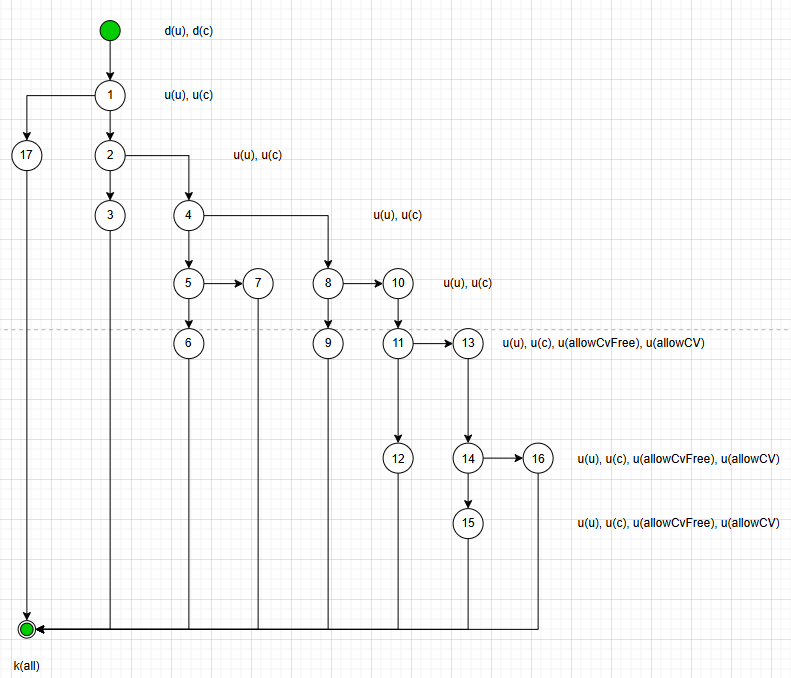
Kết quả kỳ vọng: In ra màn hình 'Ok' và giảm giá trị allowCV xuống 1.

Test case cho đường 8: 1->2->4->8->10->11->13->14->16

Value(u,c,"allowCvFree", "allowCV"):123,null,0,0

Kết quả kỳ vọng: In ra màn hình 'Công ty bạn đã hết lượt xem'.

* + 1. **Lập đồ thị dòng dữ liệu chung.**

****

* + 1. **Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Kiểm thử đời sống biến u (userId):

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuk

Kịch bản 6: ~duuuuuuuk

Kịch bản 7: ~duuuuuuuuuk

Kịch bản 8: ~duuuuuuuuuk

Kiểm thử đời sống biến c (companyId):

Kiểm thử đời sống biến u

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuk

Kịch bản 6: ~duuuuuuuk

Kịch bản 7: ~duuuuuuuuuk

Kịch bản 8: ~duuuuuuuuuk

Kiểm thử đời sống biến allowCvFree:

Kịch bản 1: ~duuuuuuuk

Kịch bản 2: ~duuuuuuuuuk

Kịch bản 3: ~duuuuuuuuuk

Kiểm thử đời sống biến allowCv:

Kịch bản 1: ~duuuuuuuk

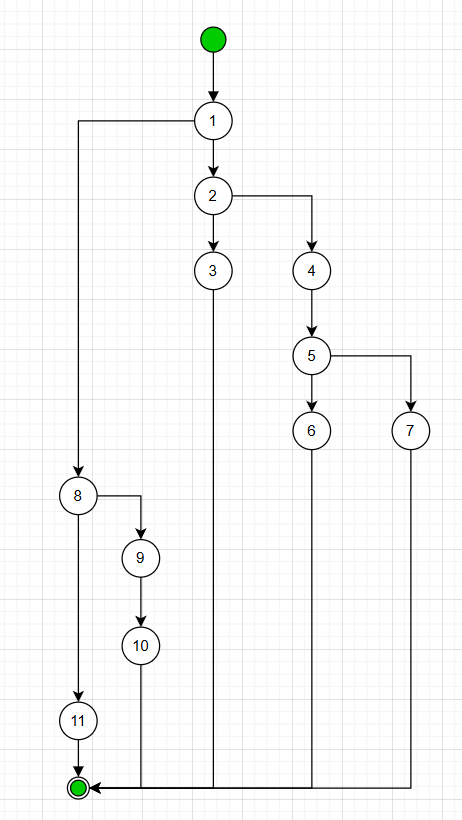
Kịch bản 2: ~duuuuuuuuuk

Kịch bản 3: ~duuuuuuuuuk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.2. Hàm creatNewPackageCv.

|  |
| --- |
| let creatNewPackageCv = (data) => {      return new Promise(async (resolve, reject) => {          try (1) {              if (!data.name || !data.price || !data.value) (2) {                  resolve({                      errCode: 1,                      errMessage: 'Missing required parameters !'                  }) (3)              } else {                  let packageCv = await db.PackageCv.create({                      name: data.name,                      value: data.value,                      price: data.price,                      isActive: 1                  }) (4)                  if (packageCv) (5) {                      resolve({                          errCode: 0,                          errMessage: 'Tạo gói sản phẩm thành công'                      }) (6)                  }                  Else {                      resolve({                          errCode: 2,                          errMessage: 'Tạo gói sản phẩm thất bại'                      }) (7)                  }              }          } catch (error) (8) {              if (error.message.includes('Validation error')) (9) {                  resolve({                      errCode: 2,                      errMessage: 'Tên gói sản phẩm đã tồn tại'                  }) (10)              }              else {                  reject(error)              } (11)          }      })  } |

1. **Lập đồ thị dòng điều khiển cơ bản.**
2. **Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 4 nút quyết định nhị phân nên có độ phức tạp C = 4+1 = 5.

Vậy có ít nhất là 5 test case có thể bao phủ 100% các nhánh

1. **Thiết kế test cases cho từng đường.**

Test case cho đường 1: 1->2->3

Value(n,p,v):

Kết quả kỳ vọng: in ra màn hình 'Missing required parameters !'

Test case cho đường 2: 1->2->4->5->6

Value(n,p,v): a,1,b

Kết quả kỳ vọng: in ra màn hình 'Tạo gói sản phẩm thành công'

Test case cho đường 3: 1->2->4->5->7

Value(n,p,v): @,#,$

Kết quả kỳ vọng: in ra màn hình 'Tạo gói sản phẩm thất bại'

Test case cho đường 4: 1->8->11

Value(n,p,v): 'Validation error'

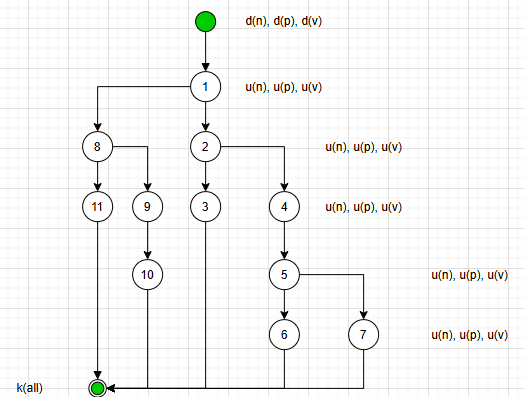
Kết quả kỳ vọng: in ra màn hình 'Tên gói sản phẩm đã tồn tại'

Test case cho đường 5: 1->8->9->10

Value(n,p,v):

Kết quả kỳ vọng: Kết thúc chương trình

1. **Lập đồ thị dòng dữ liệu chung.**

****

1. **Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Kiểm thử đời sống biến n (name):

Kịch bản 1: ~duuuk

Kịch bản 2: ~duuuuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuk

Kịch bản 5: ~duuuuk

Kiểm thử đời sống biến p (price):

Kịch bản 1: ~duuuk

Kịch bản 2: ~duuuuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuk

Kịch bản 5: ~duuuuk

Kiểm thử đời sống biến v (value):

Kịch bản 1: ~duuuk

Kịch bản 2: ~duuuuuk

Kịch bản 3: ~duuuuuk

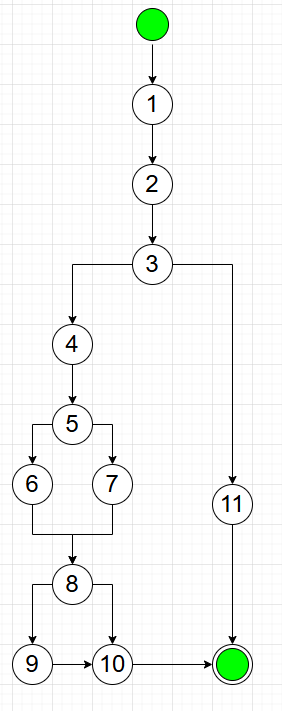
Kịch bản 4: ~duuuk

Kịch bản 5: ~duuuuk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.3. Hàm handleBanPost.

|  |
| --- |
| let handleBanPost = async (id, note) => {      let res = await banPostService({1          postId: id, 1          userId: user.id, 1          note: note 1      }) 2 3      if (res && res.errCode === 0) {          let arrData = [] 4          if (user.roleCode == 'ADMIN') { 5              arrData = await getAllPostByRoleAdminService({6                  limit: PAGINATION.pagerow, 6                  offset: numberPage \* PAGINATION.pagerow, 6                  search: CommonUtils.removeSpace(search), 6                  censorCode: censorCode6              })          } else {              arrData = await getAllPostByAdminService({7                  limit: PAGINATION.pagerow, 7                  offset: numberPage \* PAGINATION.pagerow, 7                  companyId: user.companyId, 7                  search: CommonUtils.removeSpace(search), 7                  censorCode: censorCode7              })          }          if (arrData && arrData.errCode === 0) { 8              setdataPost(arrData.data) 9              setTotal(arrData.count) 9          }          toast.success(res.errMessage) 10      } else {          toast.error(res.errMessage) 11      } |

* + 1. **Lập đồ thị dòng điều khiển cơ bản.**
    2. **Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 3 nút quyết định nhị phân nên có độ phức tạp C = 3+1 = 4.

Vậy có ít nhất là 4 test case có thể bao phủ 100% các nhánh

* + 1. **Thiết kế test cases cho từng đường.**

Test case cho đường 1: 1→2→3→4→5→6→8→9→10

Value(id, note, user.roleCode, res.errCode, arrData.errCode): 123, "Banned for violating rules", "ADMIN", 0, 0

Kết quả kỳ vọng: Cập nhật setdataPost và setTotal. In ra màn hình "Operation Successful".

Test case cho đường 2: 1→2→3→4→5→7→8→9→10

Value(id, note, user.roleCode, res.errCode, arrData.errCode): 456, "Spam post", "USER", 0, 0

Kết quả kỳ vọng: Cập nhật setdataPost và setTotal. In ra màn hình "Operation Successful".

Test case cho đường 3: 1→2→3→11

Value(id, note, res.errCode): null, null, 1

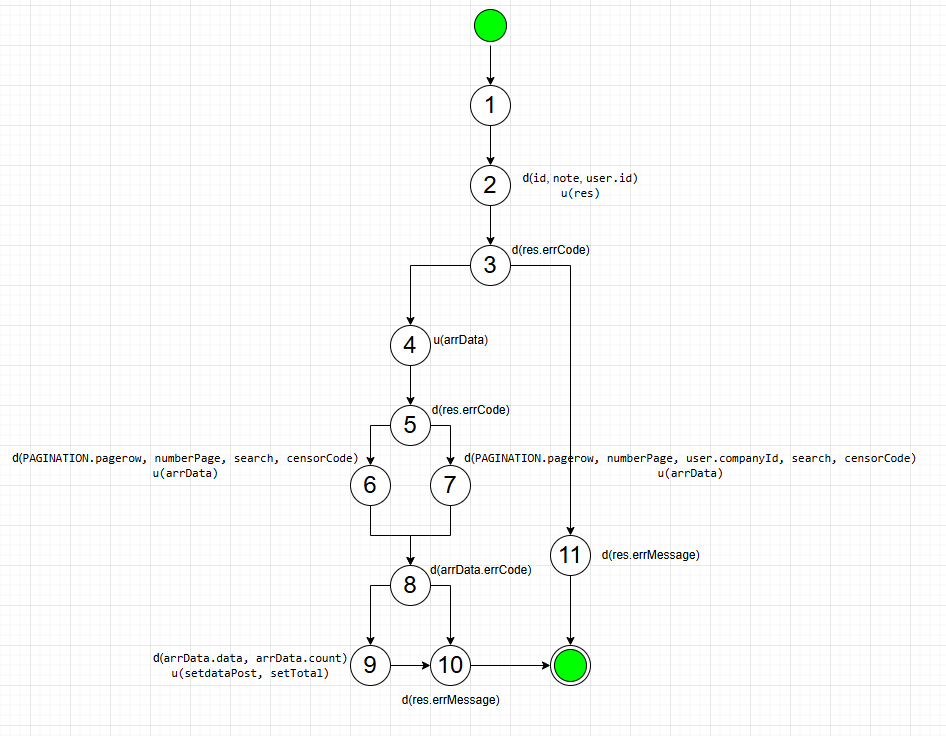
Kết quả kỳ vọng: In ra màn hình "Invalid Operation".

Test case cho đường 4: 1→2→3→4→5→6→8→10

Value(id, note, user.roleCode, res.errCode, arrData.errCode): 789, "Test ban post", "ADMIN", 0, 1

Kết quả kỳ vọng: Không cập nhật setdataPost hoặc setTotal. In ra màn hình "Operation Successful".

* + 1. **Lập đồ thị dòng dữ liệu chung.**



* + 1. **Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Kiểm thử đời sống biến id

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuuuk

Kịch bản 6: ~duuuuuuuuuk

Kiểm thử đời sống biến note

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuk

Kịch bản 6: ~duuuuuuuk

Kiểm thử đời sống biến res

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuuuk

Kịch bản 6: ~duuuuuuuuuk

Kiểm thử đời sống biến arrData

Kịch bản 1: ~duuuuuuk

Kịch bản 2: ~duuuuuuuk

Kịch bản 3: ~duuuuuuuk

Kịch bản 4: ~duuuuuuuk

Kiểm thử đời sống biến user.roleCode

Kịch bản 1: ~duuk

Kịch bản 2: ~duuuk

Kịch bản 3: ~duuuuk

Kịch bản 4: ~duuuuuk

Kịch bản 5: ~duuuuuk

Kịch bản 6: ~duuuuuuuk

Kiểm thử đời sống biến user.companyId

Kịch bản 1: ~duuuuuuuk

Kịch bản 2: ~duuuuuuuuuk

Kịch bản 3: ~duuuuuuuuuk

Kiểm thử đời sống biến search

Kịch bản 1: ~duuuuuuuk

Kịch bản 2: ~duuuuuuuk

Kịch bản 3: ~duuuuuuuk

Kiểm thử đời sống biến censorCode

Kịch bản 1: ~duuuuuuuk

Kịch bản 2: ~duuuuuuuk

Kịch bản 3: ~duuuuuuuk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.4. Hàm handleSaveJobSkill.

|  |
| --- |
| let handleSaveJobSkill = async () => {      setIsLoading(true) 1      if (isActionADD === true) { 2          let res = await createSkilleService({3              name: inputValues.name, 3              categoryJobCode: inputValues.categoryJobCode3          })          setTimeout(() => { 4              setIsLoading(false) 5              if (res && res.errCode === 0) { 6                  toast.success("Thêm kĩ năng thành công") 7                  setInputValues({8                      ...inputValues, 8                      ["name"]: '', 8                      ["categoryJobCode"]: '', 8                  })              }              else if (res && res.errCode === 2) { 9                  toast.error(res.errMessage) 10              }              else toast.error("Thêm kĩ năng thất bại") 11          }, 50);      } else {          let res = await UpdateSkillService({12              name: inputValues.name, 12              id: code, 12              categoryJobCode: inputValues.categoryJobCode12          })          setTimeout(() => { 13              setIsLoading(false) 14              if (res && res.errCode === 0) { 15                  toast.success("Cập nhật loại công việc thành công") 16              }              else if (res && res.errCode === 2) { 17                  toast.error(res.errMessage) 18              }              else toast.error("Cập nhật loại công việc thất bại") 19          }, 50);      }  } |

**Lập đồ thị dòng điều khiển cơ bản.**



**Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 5 nút quyết định nhị phân nên có độ phức tạp C = 5+1 = 6.

Vậy có ít nhất là 6 test case có thể bao phủ 100% các nhánh

**Thiết kế test cases cho từng đường.**

Test case cho đường 1: 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8

Value(name, categoryJobCode, isActionADD, res.errCode):

"Kỹ năng A", "JobCodeA", true, 0

Hiển thị thông báo: "Thêm kĩ năng thành công". Trường nhập được làm trống (name = "", categoryJobCode = "").

Test case cho đường 2: 1 → 2 → 3 → 4 → 5 → 6 → 9 → 10

Value(name, categoryJobCode, isActionADD, res.errCode, res.errMessage):

"Kỹ năng B", "JobCodeB", true, 2, "Dữ liệu đã tồn tại"

Hiển thị thông báo lỗi: "Dữ liệu đã tồn tại".

Test case cho đường 3: 1 → 2 → 3 → 4 → 5 → 6 → 9 → 11

Value(name, categoryJobCode, isActionADD, res.errCode):

"", "", true, -1

Hiển thị thông báo lỗi chung: "Thêm kĩ năng thất bại".

Test case cho đường 4: 1 → 2 → 12 → 13 → 14 → 15 → 16

Value(name, categoryJobCode, code, isActionADD, res.errCode):

"Cập nhật kỹ năng A", "JobCodeC", "12345", false, 0

Kết quả kỳ vọng: Hiển thị thông báo: "Cập nhật loại công việc thành công".

Test case cho đường 5: 1 → 2 → 12 → 13 → 14 → 15 → 17 → 18

Value(name, categoryJobCode, code, isActionADD, res.errCode, res.errMessage):

"Cập nhật kỹ năng B", "JobCodeD", "67890", false, 2, "Không tìm thấy công việc"

Kết quả kỳ vọng: Hiển thị thông báo lỗi: "Không tìm thấy công việc"

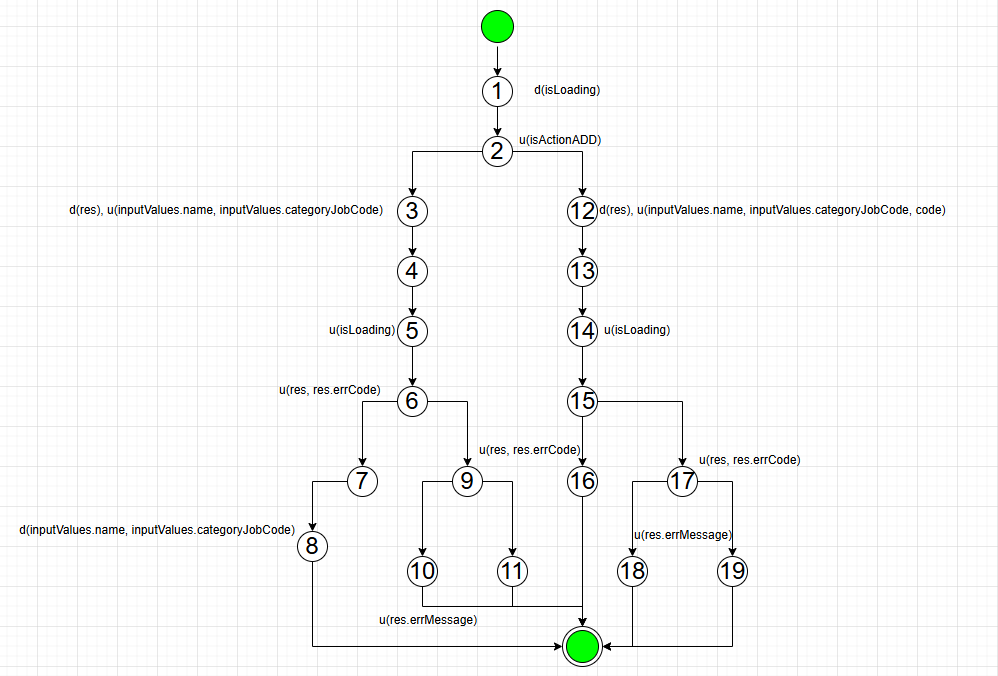
Test case cho đường 6: 1 → 2 → 12 → 13 → 14 → 15 → 17 → 19

Value(name, categoryJobCode, code, isActionADD, res.errCode):

"Cập nhật kỹ năng C", "", "98765", false, -1

Kết quả kỳ vọng: Hiển thị thông báo lỗi chung: "Cập nhật loại công việc thất bại".

**Lập đồ thị dòng dữ liệu chung.**



**Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Kiểm thử đời sống biến name (inputValues.name)

Kịch bản 1: ~duuuk

Kịch bản 2: ~duuuuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuk

Kịch bản 5: ~duuuuk

Kiểm thử đời sống biến categoryJobCode:

Kịch bản 1: ~duuuk

Kịch bản 2: ~duuuuuk

Kịch bản 3: ~duuuuuk

Kịch bản 4: ~duuuk

Kịch bản 5: ~duuuuk

Kiểm thử đời sống biến res.errCode:

Kịch bản 1: ~uuk

Kịch bản 2: ~uuk

Kịch bản 3: ~uuk

Kiểm thử đời sống biến res.errCode:

Kịch bản 1: ~uk

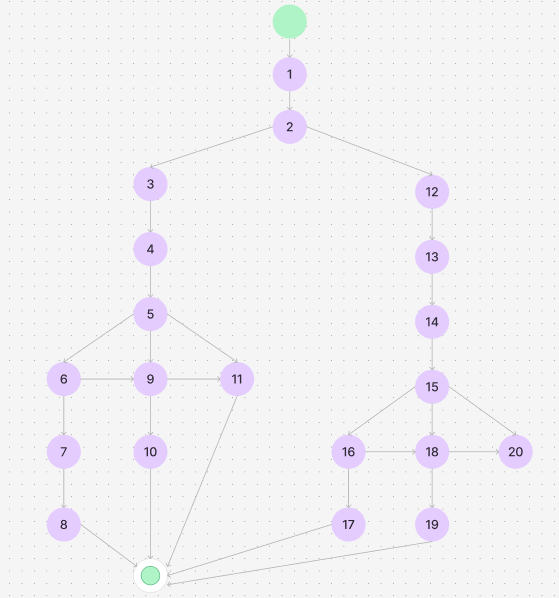
Kịch bản 2: ~uk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.5. Hàm handleSaveJobLevel.

|  |
| --- |
| let handleSaveJobLevel = async () => {  setIsLoading(true) (1)  if (isActionADD === true) { (2)  let res = await createAllCodeService({ (3)  value: inputValues.value,  code: inputValues.code,  type: 'JOBLEVEL',  })  setTimeout(() => { (4)  setIsLoading(false) (5)  if (res && res.errCode === 0) { (6)  toast.success("Thêm cấp bậc thành công") (7)  setInputValues({ (8)  ...inputValues,  ["value"]: '',  ["code"]: '',  })  }  else if (res && res.errCode === 2) { (9)  toast.error(res.errMessage) (10)  }  else toast.error("Thêm cấp bậc thất bại") (11)  }, 50);  } else { (12)  let res = await UpdateAllcodeService({ (13)  value: inputValues.value,  code: id,  })  setTimeout(() => { (14)  setIsLoading(false) (15)  if (res && res.errCode === 0) { (16)  toast.success("Cập nhật cấp bậc thành công") (17)  }  else if (res && res.errCode === 2) { (18)  toast.error(res.errMessage) (19)  }  else toast.error("Cập nhật cấp bậc thất bại") (20)  }, 50);  }  } |

**Lập đồ thị dòng điều khiển cơ bản.**



**Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 7 nút quyết định nhị phân nên có độ phức tạp C = 6+1 = 7.

Vậy có ít nhất là 7 test case có thể bao phủ 100% các nhánh

**Thiết kế test cases cho từng đường.**

Test Case 1: Đường đi 1->2->4->5->6

Điều kiện: isActionADD = true và res.errCode = 0

Input:

inputValues = { value: "Junior", code: "J001" }

Kết quả kỳ vọng:

Hiển thị thông báo thành công: "Thêm cấp bậc thành công".

inputValues được đặt lại { value: '', code: '' }.

Test Case 2: Đường đi 1->2->4->5->9

Điều kiện: isActionADD = true và res.errCode = 2

Input:

inputValues = { value: "Junior", code: "J002" }

Kết quả kỳ vọng:

Hiển thị thông báo lỗi với nội dung từ res.errMessage.

Test Case 3: Đường đi 1->2->4->5->11

Điều kiện: isActionADD = true và res.errCode !== 0 && res.errCode !== 2

Input:

inputValues = { value: "Senior", code: "S001" }

Kết quả kỳ vọng:

Hiển thị thông báo lỗi: "Thêm cấp bậc thất bại".

Test Case 4: Đường đi 1->12->13->14->16

Điều kiện: isActionADD = false và res.errCode = 0

Input:

inputValues = { value: "Manager" }

id = "M001"

Kết quả kỳ vọng:

Hiển thị thông báo thành công: "Cập nhật cấp bậc thành công".

Test Case 5: Đường đi 1->12->13->14->18

Điều kiện: isActionADD = false và res.errCode = 2

Input:

inputValues = { value: "Director" }

id = "D001"

Kết quả kỳ vọng:

Hiển thị thông báo lỗi với nội dung từ res.errMessage.

Test Case 6: Đường đi 1->12->13->14->20

Điều kiện: isActionADD = false và res.errCode !== 0 && res.errCode !== 2

Input:

inputValues = { value: "CEO" }

id = "C001"

Kết quả kỳ vọng:

Hiển thị thông báo lỗi: "Cập nhật cấp bậc thất bại".

Test Case 7: Đường đi 1

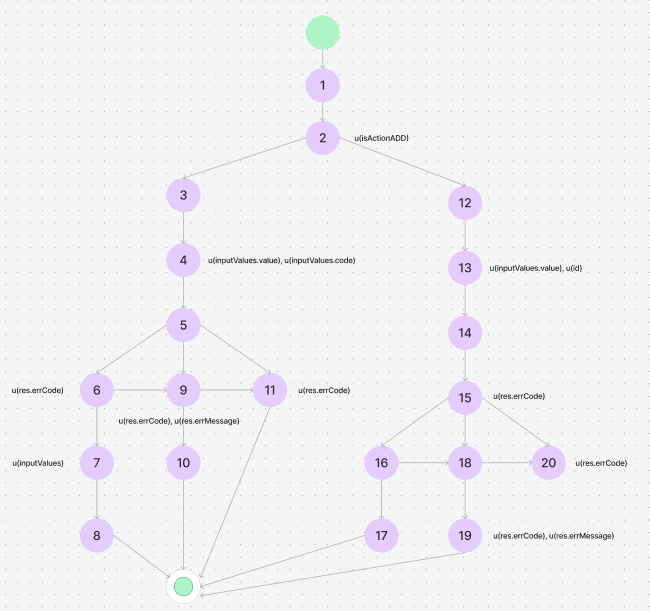
Điều kiện: setIsLoading(true) được gọi.

Input: Không có input cụ thể.

Kết quả kỳ vọng:

Biến isLoading được đặt thành true.

**Lập đồ thị dòng dữ liệu chung.**



**Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Kiểm thử đời sống biến isActionADD

* Kịch bản 1: ~duuk

Kiểm thử đời sống biến inputValues

* Kịch bản 1: ~duuuk
* Kịch bản 2: ~duuuuuk

Kiểm thử đời sống biến id

* Kịch bản 1: ~duuk

Kiểm thử đời sống biến res

* Kịch bản 1: ~duuuuuk
* Kịch bản 2: ~duuuuuk

Kiểm thử đời sống biến res.errCode

* Kịch bản 1: ~duuuuuk

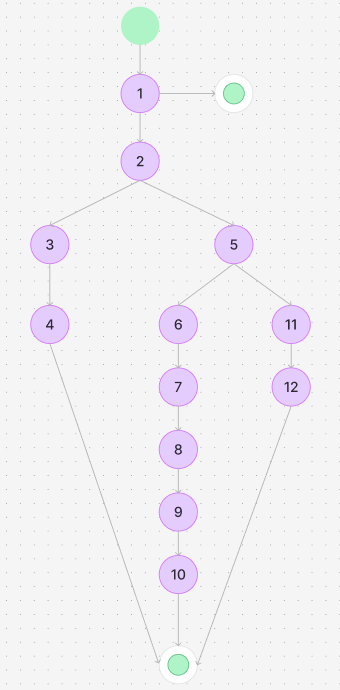
Kiểm thử đời sống biến res.errMessage

* Kịch bản 1: ~duuuuuk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.6. Hàm handleChange.

|  |
| --- |
| const handleChange = async (value, detail,type) => {  if (!value && !detail) { (1)  setInputValue({ (2)  ...inputValue,  [type]: ''  })  }  if (Array.isArray(detail)) { (3)  setInputValue({ (4)  ...inputValue,  listSkills: value  })  }  else { (5)  if (detail.type === 'categoryJobCode') { (6)  let res = await getAllSkillByJobCode(value) (7)  let listSkills = res.data.map(item => ({ (8)  value: item.id,  label: item.name  }))  setListSkills(listSkills) (9)  setInputValue({ (10)  ...inputValue,  [detail.type]: value,  listSkills: []  })  }  else {(11)  setInputValue({ (12)  ...inputValue,  [detail.type]: value,  })  }  }  }; |

**Lập đồ thị dòng điều khiển cơ bản.**

**Tính số đường thi hành tuyến tính độc lập.**

Đồ thị bên có 4 nút quyết định nhị phân nên có độ phức tạp C = 4+1 = 5.

Vậy có ít nhất là 5 test case có thể bao phủ 100% các nhánh

**Thiết kế test cases cho từng đường.**

Test case cho đường 1: 1->2

Value(value, detail, type): null, null, "jobType"

Kết quả kỳ vọng:

inputValue[type] được gán giá trị ''.

Test case cho đường 2: 1->3->4

Value(value, detail, type): ["skill1", "skill2"], ["skill1", "skill2"], "skills"

Kết quả kỳ vọng:

inputValue.listSkills được cập nhật thành ["skill1", "skill2"].

Test case cho đường 3: 1->5->6->7->8->9->10

Value(value, detail, type): "dev", { type: "categoryJobCode" }, "category"

Kết quả kỳ vọng:

Gọi API getAllSkillByJobCode với giá trị "dev".

Kết quả từ API được xử lý thành danh sách listSkills.

listSkills được cập nhật trong setListSkills.

inputValue được cập nhật với:

detail.type: "dev".

listSkills: [].

Test case cho đường 4: 1->5->11->12

Value(value, detail, type): "manager", { type: "jobLevel" }, "level"

Kết quả kỳ vọng:

inputValue được cập nhật với:

detail.type: "manager".

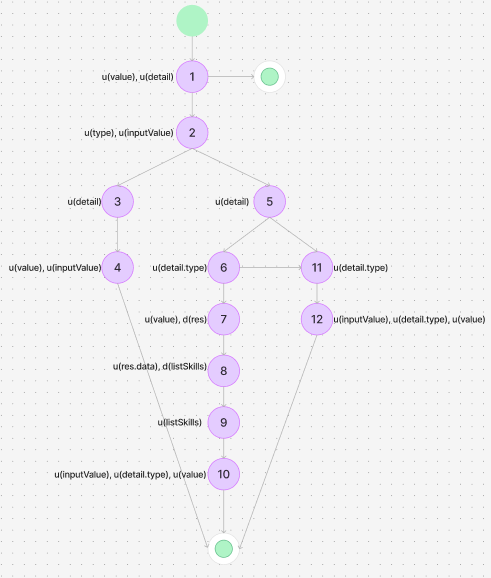
Test case cho đường 5: 1

Value(value, detail, type): undefined, undefined, undefined

Kết quả kỳ vọng:

Không thực hiện bất kỳ hành động nào (do cả value và detail đều không hợp lệ).

**Lập đồ thị dòng dữ liệu chung.**



**Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

Biến value:

* Kịch bản 1: ~duuuuuk
* Kịch bản 2: ~duuk

Biến detail:

* Kịch bản 1: ~duuuuuk
* Kịch bản 2: ~duuk

Biến type:

* Kịch bản 1: ~duuuuk
* Kịch bản 2: ~duuk

Biến inputValue:

* Kịch bản 1: ~duuuuuk
* Kịch bản 2: ~duuk

Biến res:

* Kịch bản 1: ~duuuuk
* Kịch bản 2: ~duuk

Biến listSkills:

* Kịch bản 1: ~duuuuuk
* Kịch bản 2: ~duuk

**Các kịch bản trên đều không chứa cặp đôi hoạt động bất thường nào cả.**

### 4.7. Hàm handleLogin (User).

|  |
| --- |
| let handleLogin = async (phonenumber, password) => {  let res = await handleLoginService({  phonenumber: phonenumber,  password: password,  });  if (res && res.errCode === 0) {  localStorage.setItem("userData", JSON.stringify(res.user));  localStorage.setItem("token\_user", res.token);  if (res.user.roleCode === "ADMIN" || res.user.roleCode === "EMPLOYER") {  window.location.href = "/admin/";  } else {  window.location.href = "/";  }  } else {  toast.error(res.errMessage);  }  }; |

**i. Lập đồ thị dòng điều khiển cơ bản.**

- Node 1: (Call handleLoginService): Gọi dịch vụ xác thực đăng nhập và nhận kết quả trả về (res).

- Node 2: (Check if res && res.errCode === 0): Kiểm tra xem kết quả có hợp lệ (tức là không có lỗi và mã lỗi là 0).

True (T): Nếu hợp lệ (không có lỗi), tiếp tục thực hiện bước [3].

False (F): Nếu không hợp lệ (có lỗi), đi đến bước [7] để hiển thị thông báo lỗi.

- Node 3: (Set userData and token\_user in LocalStorage): Lưu thông tin người dùng và token vào localStorage nếu đăng nhập thành công.

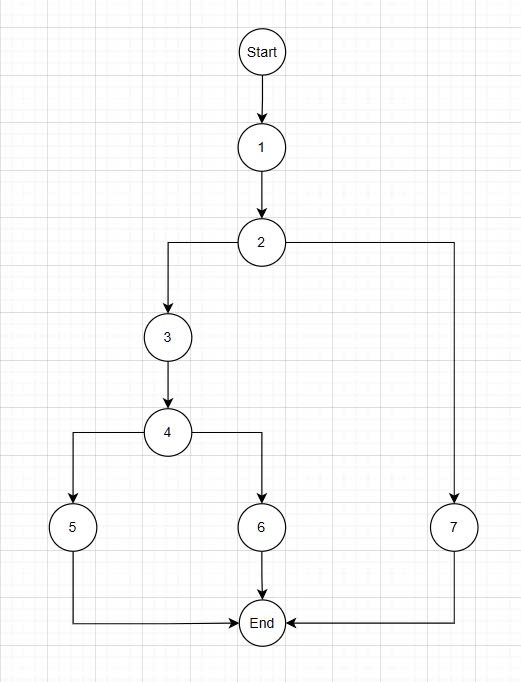
- Node 4: (Check res.user.roleCode === "ADMIN" || res.user.roleCode === "EMPLOYER"): Kiểm tra giá trị của roleCode để quyết định chuyển hướng.

True (T): Nếu người dùng có quyền ADMIN hoặc EMPLOYER, chuyển hướng đến trang quản trị (/admin/) tại bước [5].

False (F): Nếu không phải ADMIN hoặc EMPLOYER, chuyển hướng đến trang chủ (/) tại bước [6].

- Node 5: (Redirect to "/admin/"): Chuyển hướng đến trang /admin/.

- Node 6: (Redirect to "/"): Chuyển hướng về trang chủ /.

- Node 7: (Show error message): Hiển thị thông báo lỗi từ res.errMessage nếu đăng nhập không thành công.

**ii. Tính số đường thi hành tuyến tính độc lập.**

- Đồ thị bên có 1 nút quyết định nhị phân nên có độ phức tạp C = 2 + 1 = 3.

- Vậy số đường thi hành tuyến tính độc lập là: 3

**iii. Thiết kế test cases cho từng đường.**

- Test Case cho đường 1: 1 => 2 => 3 => 4 => 5

+ Value (phonenumber, password, roleCode): "123456789", "admin123", "ADMIN" hoặc là "123456789", "admin123", "EMPLOYER"

+ Kết quả kỳ vọng: userData và token\_user được lưu vào localStorage, người dùng được chuyển hướng tới trang /admin/.

- Test Case cho đường 2: 1 => 2 => 3 => 4 => 6

+ Value (phonenumber, password, roleCode): "123456789", "admin123", "USER" hoặc là "123456789", "admin123", "GUEST"

+ Kết quả kỳ vọng: userData và token\_user được lưu vào localStorage, người dùng được chuyển hướng tới trang chủ /.

- Test Case cho đường 3: 1 => 2 => 7

+ Value (phonenumber, password, roleCode): "wrongnumber", "wrongpassword", "null"

+ Kết quả kỳ vọng: Hiển thị thông báo lỗi từ res.errMessage (ví dụ: "Invalid phone number or password"), không lưu userData và token\_user vào localStorage, không có chuyển hướng trang.

**iv. Lập đồ thị dòng dữ liệu chung.**

**v. Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

- Kiểm thử đời sống biến phonenumber và password:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

- Kiểm thử đời sống biến roleCode:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

- Kiểm thử đời sống biến errMessage:

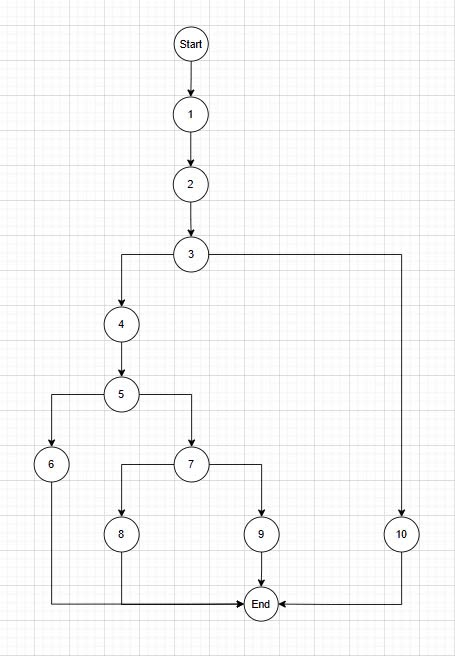
+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

### 4.8. Hàm handleLogin (Company).

|  |
| --- |
| let handleLogin = async () => {  let res = await handleLoginService({  phonenumber: inputValues.phonenumber,  password: inputValues.password  })  if (res && res.errCode === 0) {  localStorage.setItem("userData", JSON.stringify(res.user))  localStorage.setItem("token\_user", res.token)  if (res.user.roleCode === "ADMIN" || res.user.roleCode === "EMPLOYER" || res.user.roleCode === "COMPANY") {  window.location.href = "/admin/"  }  else {  const lastUrl = localStorage.getItem("lastUrl")  if (lastUrl) {  localStorage.removeItem("lastUrl")  window.location.href = lastUrl  }  else {  window.location.href = "/"  }  }  }  else {  toast.error(res.errMessage)  }  } |

**i. Lập đồ thị dòng điều khiển cơ bản.**

- Node 1: Bắt đầu hàm handleLogin.

- Node 2: Gọi hàm handleLoginService.

- Node 3: Kiểm tra điều kiện res && res.errCode === 0.

- Node 4: Lưu userData và token\_user vào localStorage.

- Node 5: Kiểm tra điều kiện roleCode (có 2 nhánh):

- Nhánh 1 (Node 6): roleCode === "ADMIN" || "EMPLOYER" || "COMPANY", chuyển hướng đến /admin/.

- Nhánh 2 (Node 7): Kiểm tra lastUrl trong localStorage.

- Node 8: Nếu có, xóa lastUrl và chuyển hướng đến lastUrl.

- Node 9: Nếu không, chuyển hướng đến /.

- Node 10: Hiển thị thông báo lỗi với toast.error.

**ii. Tính số đường thi hành tuyến tính độc lập.**

- Đồ thị bên có 3 nút quyết định nhị phân nên có độ phức tạp C = 3 + 1 = 4.

- Vậy số đường thi hành tuyến tính độc lập là: 4

**iii. Thiết kế test cases cho từng đường.**

- Test Case cho đường 1: 1 => 2 => 3 => 4 => 5 => 6

+ Value (phonenumber, password, roleCode): "123456789", "admin123", "ADMIN"

+ Kết quả kỳ vọng: Lưu userData và token\_user vào localStorage, chuyển hướng đến /admin/.

- Test Case cho đường 2: 1 => 2 => 3 => 4 => 5 => 7 => 8

+ Value (phonenumber, password, roleCode, lastUrl): "987654321", "user123", "USER", "/profile"

+ Kết quả kỳ vọng: Lưu userData và token\_user vào localStorage, chuyển hướng đến /profile.

- Test Case cho đường 3: 1 => 2 => 3 => 4 => 5 => 7 => 9

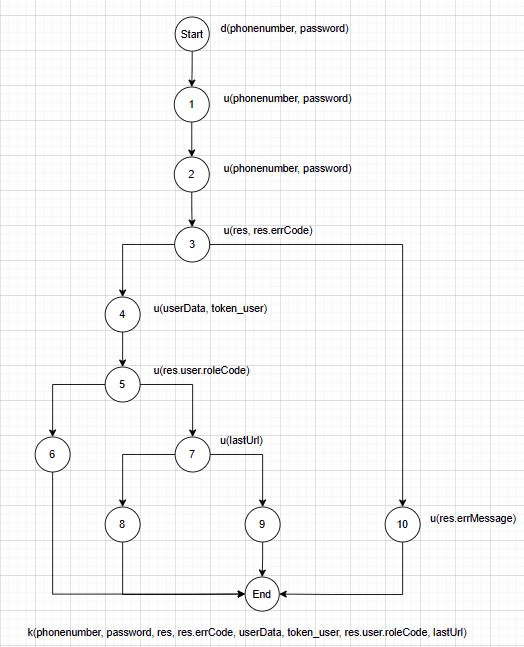
+ Value (phonenumber, password, roleCode, lastUrl): "555666777", "guest123", "GUEST", null

+ Kết quả kỳ vọng: Lưu userData và token\_user vào localStorage, chuyển hướng đến /.

- Test Case cho đường 4: 1 => 2 => 3 => 10

+ Value (phonenumber, password, errCode, errMessage): "000000000", "wrongpass", 1, "Invalid login"

+ Kết quả kỳ vọng: Hiển thị lỗi với thông báo: Invalid login, không lưu gì vào localStorage, không chuyển hướng.

**iv. Lập đồ thị dòng dữ liệu chung.**

**v.Kiểm thử đời sống của từng biến trong đơn vị mã nguồn đó.**

- Kiểm thử đời sống biến phonenumber và password:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

+ Kịch bản 4: ~duuuuk

- Kiểm thử đời sống biến userData và token\_user:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

+ Kịch bản 4: ~duuuuk

- Kiểm thử đời sống biến roleCode:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

+ Kịch bản 4: ~duuuuk

- Kiểm thử đời sống biến lastUrl:

+ Kịch bản 1: ~duk

+ Kịch bản 2: ~duuk

+ Kịch bản 3: ~duuuk

+ Kịch bản 4: ~duuuuk

## **CHƯƠNG 5: KIỂM THỬ HỘP ĐEN**

### 5.1. Tổng quan kết quả.

#### 5.1.1. Thống kê theo tính năng.

* Thực hiện kiểm thử trên 20 chức năng:

Đăng nhập ( 3 Test case ).

Tìm kiếm công việc ( 2 test case ).

Lọc công việc ( 3 Test case ).

Hiển thị thông tin hồ sơ( 1 Test case ).

Cập nhật thông tin hồ sơ ( 4 Test case ).

Ứng tuyển ( 2 Test case ).

Xem công việc đã ứng tuyển ( 1 Test case ).

Xem danh mục nghề nghiệp ( 1 Test case ).

Đăng xuất ( 2 Test case ).

Tìm kiếm công ty ( 2 Test case ).

Chia sẽ công ty ( 1 Test case ).

Quên mật khẩu ( 4 Test case ).

Xem thống kê ( 1 Test case ).

Đổi mật khẩu ( 4 Test case ).

Thay đổi phông nền ( 1 Test case ).

Đăng ký ( 10 Test case ).

Thực hiện thêm kỹ năng ( 4 Test case ).

Xem chi tiết bài đăng ( 1 Test case ).

Duyệt bài đăng ( 1 Test case ).

Từ chối bài đăng (2 Test case ).

* Trong đó có 6 chức năng lỗi:

Cập nhật thông tin hồ sơ.

Xem danh mục nghề nghiệp.

Tìm kiếm công ty, Đăng ký.

Thêm kỹ năng mới, từ chối bài đăng.

#### 5.1.2. Thống kê theo Test case.

* Thực hiện 50 Test case:

Từ FPTC\_001 đến FPTC\_050

* Trong đó có 9 Test case bị lỗi:

FPTC\_007.

FPTC\_008.

FPTC\_009.

FPTC\_010.

FPTC\_015.

FPTC\_018.

FPTC\_032.

FPTC\_045.

FPTC\_050.

Đường dẫn đến 50 Test case:

<https://docs.google.com/spreadsheets/d/1JHuy_BQ9TC4IBBXdNzhHOqx7ONVfWuhnOauzEnbsQIo/edit?gid=921290494#gid=921290494>

### 5.2. Bug Report.

#### 5.2.1. Bug Report #1 ( Test case 7 )

|  |  |
| --- | --- |
| ID number | FPTC\_007 |
| Name | Cập nhật thông tin hồ sơ với ký tự không hợp lệ |
| Reporter | Nguyễn Hồng Thông Điệp |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống cho phép lưu thông tin hồ sơ người dùng với các ký tự lạ (ví dụ: @@##$$), không hiển thị thông báo lỗi như mong đợi. |
| URL | <http://localhost:3000/candidate/info> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Critical |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Hệ thống không hiển thị thông báo lỗi khi người dùng nhập các ký tự không hợp lệ (như ký tự đặc biệt @@##$$) vào trường "Họ" và "Tên" trên trang hồ sơ. Thay vào đó, các thông tin này vẫn được lưu lại và hiển thị như bình thường.

**Steps to reproduce**

1. Người dùng đăng nhập vào hệ thống.
2. Nhấp vào biểu tượng người dùng ở góc trên bên phải và chọn "Thông tin cá nhân".
3. Tại trang thông tin cá nhân, nhập các ký tự không hợp lệ như "@@##" vào trường "Họ" và "##@@" vào trường "Tên".
4. Nhấn nút "Save" để lưu lại thay đổi.

**Expected result**

Hệ thống phải hiển thị thông báo lỗi, ví dụ: *"Các ký tự đặc biệt không được phép trong trường Họ và Tên"* và từ chối lưu dữ liệu.

**Actual result**

Hệ thống lưu thông tin với các ký tự không hợp lệ và hiển thị chúng bình thường trên giao diện.

**Notes**

Lỗi này ảnh hưởng nghiêm trọng đến tính chính xác của dữ liệu người dùng. Đề xuất kiểm tra và áp dụng quy tắc xác thực đầu vào (validation) cho các trường thông tin cá nhân.

#### 5.2.2. Bug Report #2 ( Test case 8 )

|  |  |
| --- | --- |
| ID number | FPTC\_008 |
| Name | Chức năng tải CV cho phép tệp không hợp lệ |
| Reporter | Nguyễn Hồng Thông Điệp |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống cho phép người dùng tải lên tệp không đúng định dạng (ví dụ: .txt) vào mục CV mà không hiển thị thông báo lỗi, dẫn đến việc lưu tệp không hợp lệ trong hồ sơ. |
| URL | <http://localhost:3000/candidate/usersetting> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Tính năng tải lên CV không xác thực định dạng tệp đúng quy định (chỉ hỗ trợ .pdf, .doc, .docx). Khi người dùng tải lên tệp không hợp lệ (ví dụ: example.txt), hệ thống vẫn lưu tệp và cập nhật mục CV mà không hiển thị thông báo lỗi. Điều này có thể gây ra lỗi dữ liệu hoặc gián đoạn trong quá trình xử lý hồ sơ người dùng.

**Steps to reproduce**

1. Người dùng đăng nhập vào hệ thống.
2. Nhấp vào biểu tượng người dùng ở góc trên bên phải và chọn "Thông tin cá nhân".
3. Tại trang thông tin cá nhân, nhập các ký tự không hợp lệ như "@@##" vào trường "Họ" và "##@@" vào trường "Tên".
4. Nhấn nút "Save" để lưu lại thay đổi.

**Expected result**

Hệ thống hiển thị thông báo lỗi: *"Tệp không hợp lệ. Vui lòng tải lên các tệp có định dạng .pdf, .doc, hoặc .docx."*  và từ chối lưu tệp.

**Actual result**

Hệ thống chấp nhận tệp example.txt, lưu tệp trong mục CV mà không hiển thị bất kỳ cảnh báo hay thông báo lỗi nào.

**Notes**

Lỗi này ảnh hưởng đến tính toàn vẹn dữ liệu và có thể gây ra lỗi khi người quản lý xử lý hồ sơ. Cần kiểm tra và bổ sung logic xác thực định dạng tệp trong backend và giao diện người dùng.

#### 5.2.3. Bug Report #3 ( Test case 9 )

|  |  |
| --- | --- |
| ID number | FPTC\_009 |
| Name | Chức năng chỉnh sửa ngày sinh cho phép giá trị không hợp lệ |
| Reporter | Nguyễn Hồng Thông Điệp |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống cho phép người dùng chỉnh sửa ngày sinh với giá trị không hợp lệ (ví dụ: 1/12/3000), lưu thông tin mà không hiển thị thông báo lỗi hoặc ngăn chặn. |
| URL | <http://localhost:3000/candidate/info> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Khi người dùng chỉnh sửa ngày sinh trên trang thông tin cá nhân và nhập một giá trị không hợp lệ (như ngày trong tương lai xa: 1/12/3000), hệ thống không thực hiện xác thực và vẫn lưu giá trị này. Điều này gây ra sự không chính xác trong dữ liệu hồ sơ người dùng và có thể ảnh hưởng đến các chức năng khác liên quan đến thông tin cá nhân.

**Steps to reproduce**

1. Người dùng đăng nhập vào hệ thống.
2. Truy cập vào trang thông tin cá nhân.
3. Tại trường "Ngày sinh", nhập giá trị không hợp lệ, ví dụ: 1/12/3000.
4. Nhấn nút "Save" để lưu thay đổi.

**Expected result**

Hệ thống hiển thị thông báo lỗi: *"Ngày sinh không hợp lệ. Vui lòng nhập ngày trong khoảng thời gian hợp lý."* và từ chối lưu thông tin.

**Actual result**

Hệ thống chấp nhận giá trị ngày sinh không hợp lệ, lưu thông tin và hiển thị giá trị này trong hồ sơ người dùng mà không có bất kỳ thông báo lỗi nào.

**Notes**

Lỗi này ảnh hưởng nghiêm trọng đến tính hợp lệ của dữ liệu người dùng và có thể gây ra lỗi trong các chức năng liên quan đến xử lý thông tin cá nhân. Đề xuất kiểm tra và bổ sung logic xác thực đầu vào cho trường ngày sinh, đảm bảo giới hạn trong khoảng thời gian hợp lệ.

#### 5.2.4. Bug Report #4 ( Test case 10 )

|  |  |
| --- | --- |
| ID number | FPTC\_010 |
| Name | Chức năng cập nhật hình ảnh cho phép file định dạng không hợp lệ |
| Reporter | Nguyễn Hồng Thông Điệp |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống cho phép người dùng tải lên và hiển thị file không hợp lệ (định dạng .txt) vào phần hình ảnh trên trang hồ sơ, dẫn đến lỗi từ server khi lưu thay đổi. |
| URL | <http://localhost:3000/candidate/info> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Chức năng chỉnh sửa hồ sơ người dùng cho phép chọn file .txt cho phần hình ảnh thay vì giới hạn định dạng file hợp lệ (như .jpg, .png). File được hiển thị trên giao diện mà không có bất kỳ thông báo lỗi nào. Khi bấm lưu thay đổi, server trả về lỗi do không xử lý được định dạng file không hợp lệ.

**Steps to reproduce**

1. Người dùng đăng nhập vào hệ thống.
2. Truy cập vào trang thông tin cá nhân.
3. Tại mục chỉnh sửa hình ảnh, chọn file 21110166\_HW1.txt từ máy tính.
4. Nhấn "Save" để lưu thay đổi.

**Expected result**

Hệ thống hiển thị thông báo lỗi: *"File không hợp lệ. Vui lòng tải lên tệp định dạng hình ảnh như .jpg, .png."* và từ chối hiển thị hoặc lưu file.

**Actual result**

Hệ thống chấp nhận và hiển thị file .txt trong mục hình ảnh mà không thông báo lỗi. Khi bấm "Save", server trả về lỗi không xử lý được định dạng file.

**Notes**

Lỗi này ảnh hưởng đến trải nghiệm người dùng và có thể gây ra sự cố khi lưu dữ liệu trên server. Đề xuất bổ sung logic kiểm tra định dạng file ở cả frontend và backend để ngăn người dùng tải lên tệp không hợp lệ.

#### 5.2.5. Bug Report #5 ( Test case 15 )

|  |  |
| --- | --- |
| ID number | FPTC\_015 |
| Name | Đường dẫn hỏng hoặc URL không hoạt động cho danh mục nghề nghiệp |
| Reporter | Nguyễn Hồng Thông Điệp |
| Submit Date | 21/11/2024 |
| Summary | Khi người dùng nhấp vào danh mục "Công nghệ thông tin" trên trang chủ, hệ thống không chuyển hướng đến trang việc làm tương ứng mà hiển thị màn hình trắng. |
| URL | <http://localhost:3000/job_listing.html> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Critical |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Chức năng điều hướng đến trang việc làm theo danh mục nghề nghiệp trên trang chủ không hoạt động. Khi người dùng nhấp vào mục "Công nghệ thông tin", hệ thống không tải được nội dung và hiển thị màn hình trắng. Lỗi này có thể ảnh hưởng đến trải nghiệm người dùng và làm gián đoạn quy trình tìm kiếm thông tin việc làm trên trang.

**Steps to reproduce**

1. Người dùng truy cập vào trang chủ.
2. Nhấp vào danh mục "Công nghệ thông tin".

**Expected result**

Hệ thống chuyển hướng người dùng đến trang hiển thị danh sách công việc trong lĩnh vực "Công nghệ thông tin".

**Actual result**

Hệ thống hiển thị màn hình trắng, không có nội dung hoặc thông báo lỗi nào.

**Notes**

Lỗi này có thể do đường dẫn bị hỏng hoặc thiếu file nội dung trên server. Đề xuất kiểm tra lại liên kết URL và logic xử lý backend cho danh mục "Công nghệ thông tin". Ngoài ra, cần bổ sung thông báo lỗi giao diện nếu không thể tải nội dung.

#### 5.2.6. Bug Report #6 ( Test case 18 )

|  |  |
| --- | --- |
| ID number | FPTC\_018 |
| Name | Chức năng tìm kiếm công ty không xử lý từ khóa không hợp lệ |
| Reporter | Trần Khải Hoàn |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống không hiển thị thông báo lỗi khi người dùng tìm kiếm công ty với từ khóa không hợp lệ (ví dụ: "!@#$"). Thay vào đó, hệ thống chỉ hiển thị kết quả "0 công ty được tìm thấy" mà không yêu cầu người dùng nhập lại từ khóa hợp lệ. |
| URL | <http://localhost:3000/company> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Khi người dùng nhập từ khóa không hợp lệ (chỉ bao gồm các ký tự đặc biệt như "!@#$") vào chức năng tìm kiếm công ty, hệ thống không xử lý đầu vào này một cách chính xác. Thay vì hiển thị thông báo lỗi và yêu cầu người dùng nhập lại từ khóa hợp lệ, hệ thống trả về kết quả "0 công ty được tìm thấy". Điều này làm giảm trải nghiệm người dùng, đặc biệt đối với những người không quen với việc nhập từ khóa đúng.

**Steps to reproduce**

1. Truy cập vào trang tìm kiếm công ty.
2. Nhập từ khóa không hợp lệ, ví dụ: "!@#$".
3. Nhấn nút "Tìm kiếm".

**Expected result**

Hệ thống hiển thị thông báo lỗi: *"Từ khóa không hợp lệ. Vui lòng chỉ nhập chữ cái, số hoặc các ký tự được phép như dấu cách."* và từ chối tìm kiếm với từ khóa không hợp lệ.

**Actual result**

Hệ thống hiển thị kết quả "0 công ty được tìm thấy" mà không đưa ra bất kỳ cảnh báo hoặc hướng dẫn nào cho người dùng.

**Notes**

Lỗi này ảnh hưởng đến tính năng tìm kiếm và gây khó khăn cho người dùng. Đề xuất thêm logic kiểm tra và xác thực từ khóa đầu vào ở frontend và backend để hướng dẫn người dùng nhập đúng định dạng. Đồng thời, cải thiện giao diện với thông báo lỗi rõ ràng và dễ hiểu.

#### 5.2.7. Bug Report #7 ( Test case 32 )

|  |  |
| --- | --- |
| ID number | FPTC\_032 |
| Name | Chức năng đăng ký không xử lý ký tự không hợp lệ |
| Reporter | Đoàn Thái Sơn |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống không hiển thị thông báo lỗi khi người dùng nhập ký tự không hợp lệ (ví dụ: @@## , ,##@@) vào các trường thông tin đăng ký. Thay vào đó, hệ thống hoàn tất và chuyển người dùng đến bước xác nhận thông tin. |
| URL | <http://localhost:3000/register> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Chức năng đăng ký không kiểm tra và xử lý đầu vào khi người dùng nhập ký tự không hợp lệ (chẳng hạn như ký tự đặc biệt @@##,, ,##@@). Hệ thống không hiển thị thông báo lỗi hoặc yêu cầu người dùng nhập lại thông tin hợp lệ. Thay vào đó, hệ thống hoàn tất và chuyển người dùng đến bước xác nhận thông tin. Điều này làm giảm tính chính xác của dữ liệu người dùng và có thể gây ra sự cố trong quá trình đăng ký.

**Steps to reproduce**

1. Truy cập vào trang đăng ký.
2. Nhập thông tin vào các trường yêu cầu (ví dụ: Họ: @@##$$, Tên: $$##@@).
3. Nhấn nút "Đăng ký".

**Expected result**

Hệ thống phải hiển thị thông báo lỗi rõ ràng: *"Ký tự không hợp lệ. Vui lòng nhập lại thông tin hợp lệ."* và yêu cầu người dùng nhập lại thông tin đúng định dạng.

**Actual result**

Hệ thống không hiển thị thông báo lỗi mà hoàn tất đăng ký và chuyển người dùng đến bước xác nhận thông tin.

**Notes**

Lỗi này ảnh hưởng đến tính toàn vẹn của dữ liệu người dùng và có thể gây ra sự cố trong quá trình xác thực hoặc xử lý thông tin. Đề xuất thêm logic xác thực đầu vào ở phía frontend và backend để xử lý ký tự không hợp lệ và yêu cầu người dùng nhập lại thông tin đúng định dạng.

#### 5.2.8. Bug Report #8 ( Test case 45 )

|  |  |
| --- | --- |
| ID number | FPTC\_045 |
| Name | Chức năng thêm kỹ năng không xử lý ký tự số trong tên kỹ năng |
| Reporter | Phạm Ngọc Đăng Khoa |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống không xử lý việc thêm kỹ năng với tên kỹ năng là ký tự số (ví dụ: "1"). Mặc dù hệ thống yêu cầu tên kỹ năng phải là chuỗi ký tự, nhưng vẫn cho phép thêm kỹ năng thành công với tên kỹ năng là số. |
| URL | <http://localhost:3000/admin/add-job-skill/> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Chức năng thêm kỹ năng không kiểm tra và xử lý việc nhập tên kỹ năng với ký tự không hợp lệ, ví dụ như ký tự số ("1"). Dù hệ thống yêu cầu tên kỹ năng phải là một chuỗi ký tự, hệ thống vẫn cho phép người dùng thêm kỹ năng với tên là số mà không hiển thị thông báo lỗi. Điều này có thể dẫn đến tình trạng không đồng nhất trong dữ liệu kỹ năng và làm giảm tính chính xác của thông tin.

**Steps to reproduce**

1. Đăng nhập vào hệ thống với tài khoản Admin.
2. Chọn "Thêm kỹ năng" trong phần "Quản lý kỹ năng".
3. Nhập tên kỹ năng là "1", chọn lĩnh vực "Truyền thông" và nhấn "Lưu".

**Expected result**

Hệ thống phải hiển thị thông báo lỗi: *"Tên kỹ năng không hợp lệ. Vui lòng nhập tên kỹ năng là chữ cái hoặc ký tự hợp lệ."* và yêu cầu người dùng nhập lại tên kỹ năng đúng định dạng.

**Actual result**

Hệ thống không hiển thị thông báo lỗi mà vẫn cho phép thêm kỹ năng với tên "1" và lưu thành công vào hệ thống.

**Notes**

Lỗi này ảnh hưởng đến tính toàn vẹn của dữ liệu kỹ năng và có thể gây khó khăn trong việc quản lý sau này. Đề xuất kiểm tra và bổ sung logic xác thực đầu vào cho trường tên kỹ năng để đảm bảo tính chính xác và hợp lệ của dữ liệu.

#### 5.2.9. Bug Report #9 ( Test case 50 )

|  |  |
| --- | --- |
| ID number | FPTC\_050 |
| Name | Chức năng từ chối bài đăng không yêu cầu gửi lý do khi từ chối |
| Reporter | Phạm Ngọc Đăng Khoa |
| Submit Date | 21/11/2024 |
| Summary | Hệ thống cho phép Admin từ chối bài đăng mà không yêu cầu nhập lý do, mặc dù hệ thống yêu cầu phải có lý do khi từ chối bài đăng. |
| URL | <http://localhost:3000/admin/list-post-admin/> |
| Screenshot |  |
| Platform | Web |
| Operating System | Windows 10 |
| Browser | Google Chrome 131.0.6778.13 |
| Severity | Major |
| Assigned to | Trần Khải Hoàn |
| Priority | High |

**Description**

Khi Admin từ chối bài đăng trong trạng thái "Chờ kiểm duyệt", hệ thống yêu cầu nhập lý do từ chối, nhưng nếu Admin không nhập lý do và nhấn "Hoàn thành", hệ thống vẫn thông báo từ chối bài đăng thành công mà không yêu cầu lý do. Điều này có thể gây ra sự thiếu minh bạch và không đảm bảo tính chính xác trong quy trình kiểm duyệt bài đăng.

**Steps to reproduce**

1. Đăng nhập vào hệ thống bằng tài khoản Admin.
2. Chọn "Danh sách bài đăng" trong mục "Quản lý bài đăng".
3. Lọc bài đăng ở trạng thái "Chờ kiểm duyệt".
4. Nhấn "Từ chối" ở bài đăng muốn từ chối.
5. Không nhập lý do và nhấn "Hoàn thành".

**Expected result**

Hệ thống phải yêu cầu Admin nhập lý do từ chối trước khi có thể hoàn thành quá trình từ chối bài đăng.

**Actual result**

Hệ thống thông báo từ chối bài đăng thành công mà không yêu cầu lý do.

**Notes**

Lỗi này ảnh hưởng đến quy trình kiểm duyệt và có thể dẫn đến việc thiếu thông tin cho nhà tuyển dụng khi bài đăng bị từ chối. Đề xuất kiểm tra lại logic yêu cầu nhập lý do trước khi từ chối bài đăng và bổ sung thông báo xác nhận lý do từ chối.

## **CHƯƠNG 6: HỆ THỐNG QUẢN LÝ BUG**

### 6.1. Giới thiệu hệ thống Jira.

Jira là một công cụ quản lý dự án và theo dõi lỗi hàng đầu, được phát triển bởi Atlassian, giúp các nhóm làm việc quản lý công việc một cách hiệu quả và chính xác. Ban đầu, Jira được thiết kế để theo dõi lỗi phần mềm, nhưng theo thời gian, hệ thống này đã mở rộng phạm vi ứng dụng sang nhiều lĩnh vực như quản lý dự án Agile, DevOps, ITSM (quản lý dịch vụ CNTT), và nhiều ngành khác. Jira cho phép các nhóm định nghĩa quy trình làm việc, phân công nhiệm vụ, theo dõi tiến độ, và phân tích hiệu suất. Với khả năng tùy chỉnh mạnh mẽ, Jira có thể đáp ứng nhu cầu của mọi nhóm làm việc, từ các nhóm phát triển nhỏ đến các tổ chức lớn. Đặc biệt, Jira hỗ trợ tích hợp với các công cụ phổ biến như Confluence, Bitbucket, Slack, giúp tăng cường tính liên kết và hiệu quả trong công việc.

### 6.2. Các tính năng nổi bật.

#### 6.2.1. Quản lý lỗi và vấn đề (Issue Tracking).

Một trong những tính năng cốt lõi của Jira là khả năng quản lý lỗi và vấn đề trong các dự án. Người dùng có thể theo dõi toàn bộ vòng đời của một lỗi hoặc vấn đề, từ khi nó được phát hiện đến khi giải quyết xong. Mỗi lỗi được định nghĩa là một "issue" với đầy đủ thông tin như tiêu đề, mô tả, mức độ ưu tiên, người chịu trách nhiệm, và trạng thái. Jira cung cấp hệ thống trạng thái linh hoạt, có thể tùy chỉnh để phù hợp với từng dự án, như "Mới", "Đang xử lý", "Đã hoàn thành". Ngoài ra, các tính năng như gắn nhãn (labels), đính kèm tài liệu, và liên kết với các lỗi khác giúp việc quản lý vấn đề trở nên trực quan và dễ dàng hơn.

#### 6.2.2. Hỗ trợ Agile và Scrum/Kanban.

Jira là một công cụ lý tưởng cho các nhóm làm việc theo phương pháp Agile. Với các bảng Scrum, Jira giúp quản lý backlog, lập kế hoạch sprint, và theo dõi tiến độ của nhóm thông qua các biểu đồ Burndown, Velocity. Ngoài ra, bảng Kanban trong Jira cho phép các nhóm trực quan hóa luồng công việc, quản lý tài nguyên hiệu quả và giảm thiểu tắc nghẽn. Jira hỗ trợ các tính năng như Epic, User Stories, và Tasks, giúp nhóm phát triển dễ dàng theo dõi tiến độ công việc ở nhiều cấp độ khác nhau. Các công cụ này không chỉ giúp nhóm lập kế hoạch mà còn đánh giá hiệu suất để cải tiến liên tục.

#### 6.2.3. Báo cáo và phân tích (Reporting & Analytics).

Jira cung cấp hệ thống báo cáo và phân tích mạnh mẽ với các biểu đồ và báo cáo thời gian thực, giúp các nhóm hiểu rõ hiệu suất và tiến độ công việc. Người dùng có thể tạo các báo cáo chi tiết như biểu đồ Burndown để theo dõi việc hoàn thành công việc trong sprint, biểu đồ Velocity để đánh giá năng suất của nhóm, hay các báo cáo trạng thái dự án nhằm cung cấp cái nhìn tổng quan về tiến độ. Các báo cáo này có thể tùy chỉnh để đáp ứng nhu cầu của từng nhóm hoặc dự án, giúp người quản lý nhanh chóng phát hiện vấn đề và đưa ra các quyết định kịp thời.

#### 6.2.4. Tích hợp hệ thống.

Một điểm mạnh khác của Jira là khả năng tích hợp với hơn 3000 công cụ khác nhau như Confluence, Bitbucket, GitHub, Slack, Jenkins, và nhiều công cụ CI/CD khác. Những tích hợp này giúp đồng bộ hóa công việc giữa các nhóm và giảm thiểu sự gián đoạn. Ngoài ra, API của Jira rất mạnh mẽ, cho phép các nhà phát triển xây dựng các tích hợp tùy chỉnh để phù hợp với quy trình làm việc cụ thể. Điều này giúp Jira trở thành một nền tảng trung tâm trong hệ sinh thái công cụ của tổ chức.

#### 6.2.5. Quy trình làm việc tùy chỉnh (Custom Workflows).

Jira cho phép người dùng tùy chỉnh quy trình làm việc (workflow) để đáp ứng yêu cầu cụ thể của từng dự án hoặc tổ chức. Người dùng có thể định nghĩa các trạng thái (status) và các bước chuyển đổi (transitions) giữa các trạng thái này. Ngoài ra, hệ thống còn hỗ trợ thêm các quy tắc tự động hóa như gán công việc tự động hoặc thông báo khi một trạng thái thay đổi. Tính năng này giúp các nhóm thiết kế quy trình làm việc rõ ràng, từ đó tăng tính minh bạch và hiệu quả.

### 6.3. Cách thức cài đặt và cấu hình.

#### 6.3.1. Cài đặt hệ thống.

Cài đặt Jira phụ thuộc vào lựa chọn giữa hai phiên bản: Jira Cloud hoặc Jira Server. Jira Cloud không yêu cầu phần cứng, người dùng chỉ cần đăng ký và sử dụng qua trình duyệt. Đối với Jira Server, hệ thống yêu cầu một máy chủ riêng với cấu hình tối thiểu là 4GB RAM, 10GB dung lượng đĩa trống, và cơ sở dữ liệu hỗ trợ như PostgreSQL, MySQL hoặc Oracle. Để cài đặt Jira Cloud, người dùng truy cập trang chủ Atlassian, tạo tài khoản và làm theo hướng dẫn. Trong khi đó, cài đặt Jira Server yêu cầu tải phần mềm từ Atlassian, cài đặt trên máy chủ, và cấu hình các thông số như cơ sở dữ liệu, kết nối mạng. Cả hai phiên bản đều có hướng dẫn chi tiết trên trang chủ Atlassian, giúp người dùng dễ dàng triển khai hệ thống.

#### 6.3.2. Cấu hình hệ thống.

Sau khi cài đặt, người dùng cần cấu hình Jira để phù hợp với nhu cầu của tổ chức. Việc đầu tiên là tạo dự án và lựa chọn loại dự án như Scrum, Kanban, hoặc Business. Tiếp theo, người dùng có thể định nghĩa các thành phần (components) và nhóm làm việc trong từng dự án. Ngoài ra, việc tùy chỉnh quy trình làm việc (workflow) rất quan trọng. Người dùng có thể vào phần Settings > Issues > Workflows để tạo hoặc chỉnh sửa quy trình với các trạng thái và bước chuyển đổi phù hợp. Cuối cùng, người quản trị cần thêm thành viên, phân quyền và thiết lập bảo mật để đảm bảo dữ liệu được bảo vệ.

### 6.4. Ưu điểm và nhược điểm.

#### 6.4.1. Ưu điểm

Jira là một công cụ mạnh mẽ và linh hoạt, cung cấp giao diện trực quan và dễ sử dụng. Hệ thống hỗ trợ nhiều tính năng hữu ích như quản lý lỗi, tích hợp Agile, và khả năng tùy chỉnh quy trình làm việc. Với khả năng tích hợp đa dạng, Jira phù hợp với nhiều tổ chức và nhóm làm việc khác nhau, từ các đội phát triển phần mềm nhỏ đến các doanh nghiệp lớn. Đặc biệt, hệ thống báo cáo thời gian thực và phân tích chi tiết giúp người quản lý dễ dàng theo dõi tiến độ và hiệu suất của nhóm.

#### 6.4.2. Nhược điểm

Tuy nhiên, Jira cũng có một số nhược điểm. Đối với các tổ chức nhỏ hoặc nhóm có ngân sách hạn chế, chi phí sử dụng Jira có thể là một rào cản, đặc biệt với các gói trả phí. Ngoài ra, vì Jira cung cấp quá nhiều tính năng, việc làm quen và cấu hình ban đầu có thể mất nhiều thời gian, đặc biệt đối với những người mới sử dụng. Một số người dùng cũng nhận thấy giao diện có thể trở nên phức tạp khi quản lý nhiều dự án lớn.

### 6.5. Kết luận về Jira.

Jira là một công cụ quản lý dự án và lỗi hiệu quả, giúp các nhóm làm việc tổ chức công việc, theo dõi tiến độ, và cải thiện hiệu suất. Với khả năng tùy chỉnh linh hoạt, tích hợp mạnh mẽ, và hỗ trợ cho phương pháp Agile, Jira phù hợp cho nhiều tổ chức khác nhau. Tuy nhiên, việc triển khai hệ thống cần đầu tư cả về thời gian và chi phí để đạt hiệu quả tối ưu. Jira không chỉ là một công cụ mà còn là một giải pháp toàn diện cho quản lý dự án và theo dõi lỗi.

### 6.6. Hình ảnh minh chứng Bug Report trên Jira.

Sau khi khởi tạo dự án Jira, chúng em tạo các Task và phân loại dựa trên trạng thái (TO DO, IN PROGRESS, DONE). Tiếp đến, chúng em tạo các Sub Task trong Task GK-7 (Lựa chọn và cài đặt một hệ thống quản lý lỗi), các Sub Task này chính là các Bug Report để giúp nhóm quản lý lỗi một cách hiệu quả hơn.

Sau đây là hình ảnh giao diện chính dự án Jira của nhóm chúng em:

|  |
| --- |
|  |
|  |

Tiếp theo là giao diện khi bấm vào Task GK-7 (Lựa chọn và cài đặt một hệ thống quản lý lỗi), bên trong chứa các Bug Report để thuận tiện cho việc quản lý lỗi:

|  |
| --- |
|  |
|  |

# PHẦN KẾT LUẬN

1. **Những điều đã đạt được trong đề tài**

Trong quá trình thực hiện đề tài này, chúng em đã đạt được những điều quan trọng sau:

* Kiến thức học được:
* Phân lớp tương đương: Chúng em đã tìm hiểu và áp dụng phương pháp phân lớp tương đương, giúp đơn giản hóa việc kiểm thử bằng cách chia các trường hợp đầu vào thành các nhóm có cùng đặc điểm.
* Các mức độ và phương pháp kiểm thử: Tìm hiểu rõ lý thuyết về các mức độ kiểm thử (đơn vị, tích hợp, hệ thống, chấp nhận,...) và các phương pháp kiểm thử như hộp trắng, hộp đen, từ đó áp dụng hiệu quả vào đề tài.
* Kỹ thuật kiểm thử hộp đen và hộp trắng: Học được 8 kỹ thuật kiểm thử hộp đen để thiết kế test case, cũng như áp dụng kiểm thử hộp trắng để kiểm tra dòng điều khiển và dòng dữ liệu của mã nguồn.
* Kỹ năng chuyên môn:
* Quy trình kiểm thử phần mềm: Nắm được quy trình đầy đủ từ đặc tả yêu cầu, lập kế hoạch kiểm thử (Test Plan), thiết kế test case, đến thực hiện kiểm thử và quản lý lỗi.
* Đặc tả và mô hình hóa yêu cầu: Lập được lược đồ chức năng và mô tả chi tiết các chức năng chính của website tuyển dụng và tìm việc.
* Quản lý lỗi: Hiểu rõ cách viết Bug Report và sử dụng công cụ Jira để theo dõi, quản lý bug một cách có tổ chức và hiệu quả.
* Kiểm thử tự động: Biết cách sử dụng công cụ kiểm thử tự động Selenium, giúp tăng tốc độ và tính chính xác trong quá trình kiểm thử.
* Thiết kế test case chuyên nghiệp: Thiết kế được bộ 50 test case bằng phương pháp kiểm thử hộp đen và kiểm thử 8 hàm trong dự án bằng phương pháp hộp trắng, đảm bảo độ bao phủ và phát hiện lỗi hiệu quả.
* Phân tích mã nguồn: Lập đồ thị dòng điều khiển cơ bản, tính số đường thi hành tuyến tính độc lập, và kiểm thử đời sống của từng biến trong đơn vị mã nguồn.
* Kỹ năng mềm:
* Làm việc nhóm: Rèn luyện khả năng giao tiếp, hợp tác với các thành viên trong nhóm, chia sẻ ý tưởng, phản biện và giải quyết vấn đề một cách hiệu quả.
* Quản lý thời gian: Thực hành phân chia công việc hợp lý, biết cách ưu tiên nhiệm vụ quan trọng và hỗ trợ các thành viên khác để đảm bảo tiến độ chung.
* Kỹ năng quản lý dự án: Tích lũy kinh nghiệm quản lý các giai đoạn của dự án, từ lên kế hoạch đến thực thi và đánh giá.
* Sản phẩm kiểm thử phần mềm:
* Lược đồ chức năng và đặc tả yêu cầu: Đã lập được lược đồ chức năng và mô tả chi tiết các chức năng chính của hệ thống.
* Kế hoạch kiểm thử: Xây dựng và thực hiện kế hoạch kiểm thử cho dự án, đảm bảo bao phủ các trường hợp quan trọng.
* Thiết kế test case: Hoàn thiện bộ 50 test case cho hệ thống, phát hiện và sửa lỗi hiệu quả.
* Kiểm thử hộp trắng: Thực hiện kiểm thử trên 8 hàm của hệ thống, tính toán và kiểm tra dòng điều khiển, dòng dữ liệu của mã nguồn.
* Quản lý bug: Sử dụng Jira để theo dõi và quản lý bug trong quá trình kiểm thử, đảm bảo quá trình sửa lỗi rõ ràng và minh bạch.

Những thành tựu này là kết quả của sự nỗ lực và làm việc chăm chỉ của cả nhóm, đồng thời được hỗ trợ và góp ý từ giảng viên hướng dẫn trong quá trình học môn kiểm thử phần mềm.

1. **Ưu và nhược điểm của đề tài**

**2.1. Ưu điểm**

* Quy trình kiểm thử toàn diện và thực tế: Nhóm không chỉ dừng lại ở lý thuyết mà đã áp dụng quy trình kiểm thử bài bản, từ lập kế hoạch kiểm thử (Test Plan), thiết kế Test Case, thực hiện kiểm thử đến báo cáo lỗi, giúp dự án mang tính chuyên nghiệp cao.
* Áp dụng thành công các phương pháp kiểm thử phần mềm: Áp dụng đa dạng các phương pháp kiểm thử đã được học.
* Cải thiện trải nghiệm người dùng: Việc áp dụng các phương pháp kiểm thử đã giúp phát hiện và sửa chữa các lỗi giao diện, tăng cường tính ổn định và mượt mà trong quá trình sử dụng. Điều này đảm bảo rằng người dùng có trải nghiệm tốt hơn khi tìm kiếm và ứng tuyển công việc.
* Sử dụng công cụ quản lý bug chuyên nghiệp: Việc sử dụng các công cụ quản lý bug giúp quá trình kiểm thử trở nên có tổ chức, hiệu quả và minh bạch, dễ dàng theo dõi tiến độ khắc phục lỗi.
* Test Case rõ ràng và đầy đủ: Các test case được xây dựng theo template chuẩn, giúp phát hiện và xử lý nhiều tình huống có thể xảy ra trên trang web.
* Nâng cao khả năng hỗ trợ kỹ thuật: Việc cung cấp báo cáo lỗi chi tiết kèm theo ảnh chụp màn hình không chỉ hỗ trợ nhóm phát triển nhanh chóng xử lý mà còn giúp dễ dàng trao đổi với các thành viên khác trong nhóm về tình trạng và giải pháp cho lỗi.

**2.2. Nhược điểm**

* Chưa áp dụng toàn bộ các phương pháp kiểm thử.
* Chưa hoàn thiện toàn bộ Unit Test.
* Khả năng kiểm thử tự động còn hạn chế.

1. **Hướng phát triển đề tài trong tương lai**

Trong tương lai, chúng em đề xuất một số hướng phát triển tiềm năng cho dự án:

* Tiếp tục áp dụng và mở rộng các phương pháp kiểm thử chưa hoàn thiện để đảm bảo hệ thống hoạt động ổn định và an toàn trong môi trường thực tế.
* Hoàn thành việc viết Unit Test cho tất cả các module trong dự án, đảm bảo từng thành phần nhỏ nhất của hệ thống đều hoạt động đúng chức năng và ổn định.
* Tăng cường sử dụng các công cụ kiểm thử tự động để kiểm thử với các kịch bản phức tạp, đồng thời cải thiện khả năng kiểm thử trên nhiều nền tảng khác nhau (mobile, tablet, desktop).
* Phát triển và tích hợp hệ thống báo cáo tự động để theo dõi kết quả kiểm thử, giúp quản lý dễ dàng đưa ra các quyết định cải tiến dựa trên dữ liệu kiểm thử thực tế.

# TÀI LIỆU THAM KHẢO

[1] Các loại Test Level trong Kiểm thử phần mềm, truy cập 02/11/2024

https://viblo.asia/p/cac-loai-test-level-trong-kiem-thu-phan-mem-E375zLaJZGW

[2] Black Box Testing là gì? Mách bạn cách tận dụng kỹ thuật kiểm thử hộp đen hiệu quả, truy cập 04/11/2024

https://fptshop.com.vn/tin-tuc/danh-gia/black-box-testing-175829

[3 White box testing là gì? Đánh giá quá trình kiểm thử hộp trắng và cách áp dụng cơ bản, truy cập 04/11/2024

https://fptshop.com.vn/tin-tuc/danh-gia/white-box-testing-la-gi-174647

[4] Tài liệu kiểm thử phần mềm của tác giả Nguyễn Văn Hiệp, truy cập 10/11/2024

[5] Ebook Software Testing, Paul C.Jorgensen, truy cập 12/11/2024

https://utexlms.hcmute.edu.vn/pluginfile.php/269383/mod\_resource/content/2/Software%20Testing.pdf

[6] Bài giảng Test Case Concepts & Template, FPT Software, truy cập 12/11/2024

[7] Report and resolve bugs 10x faster, truy cập 16/11/2024

https://marker.io/

[8] Jira Software – Tìm hiểu tính năng và cách sử dụng JIRA, truy cập 23/11/2024

https://topdev.vn/blog/jira-la-gi/

[9] HƯỚNG DẪN CÁCH SỬ DỤNG JIRA ĐƠN GIẢN NHẤT CHO DEVELOPER, truy cập 23/11/2024

https://viblo.asia/p/huong-dan-cach-su-dung-jira-don-gian-nhat-cho-developer-6J3ZgOJqZmB

# BẢNG PHÂN CHIA CÔNG VIỆC.

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Mã số sinh viên** | **Công việc** |
| Nguyễn Hồng Thông Điệp | 21110166 | * Lên kế hoạch cho dự án. * Lập kế hoạch kiểm thử. * Tham gia lựa chọn đề tài, công nghệ. * Chọn mã nguồn và thực hiện kiểm thử hộp trắng. * Thiết kế 15 Test Case. * Hoàn thành báo cáo. |
| Đoàn Thái Sơn | 21110289 | * Tham gia lựa chọn đề tài, công nghệ. * Chọn mã nguồn và thực hiện kiểm thử hộp trắng. * Đặc tả và mô hình hóa yêu cầu bài toán. * Thiết kế 10 Test Case. |
| Trần Khải Hoàn | 21110827 | * Tham gia lựa chọn đề tài, công nghệ. * Chọn mã nguồn và thực hiện kiểm thử hộp trắng. * Thiết kế 15 Test Case. * Lựa chọn và cài đặt hệ thống quản lý lỗi. * Viết Bug Report. |
| Phạm Ngọc Đăng Khoa | 21110214 | * Tham gia lựa chọn đề tài, công nghệ. * Chọn mã nguồn và thực hiện kiểm thử hộp trắng. * Đặc tả và mô hình hóa yêu cầu bài toán. * Thiết kế 10 Test Case. * Trình bày về các phương pháp kiểm thử đã áp dụng. |