3.3

Đầu tiên, chúng ta cùng nhìn lại các câu hỏi vấn đề đã được nêu ra lúc ban đầu. Sau đó, sẽ tiến hành đưa ra các giả thuyết để làm rõ nội dung thực hiện được phương pháp này. Cuối cùng, sẽ đánh giá thực nghiệm để đánh giá hiệu quả. Các câu hỏi đã nêu ra như sau:

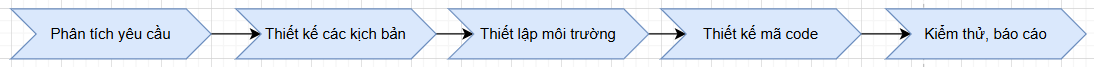
* RQ1: Vì sao việc sử dụng POM là cần thiết?
* RQ2: Làm thế nào để triển khai kiểm thử tự động bằng việc dùng POM vvới ngôn ngữ Python
* RQ3: Hiệu quả so với không dùng POM được thể hiện như thế nào?

Và sau đây sẽ là đề xuất đưa ra hướng giải quyết vấn đề:

|  |  |
| --- | --- |
| Câu hỏi nghiên cứu (RQ) | Giả thuyết (Hypothesis) |
| **RQ1**: Vì sao việc sử dụng POM là cần thiết? | Sử dụng POM giúp cải thiện khả năng bảo trì, tái sử dụng mã kiểm thử và tăng hiệu suất phát triển các kịch bản kiểm thử tự động so với cách triển khai trực tiếp. |
| **RQ2**: Làm thế nào để triển khai kiểm thử tự động bằng POM với Python? | Việc sử dụng thư viện như Selenium kết hợp với cách tổ chức mã theo POM có thể giúp xây dựng các bài kiểm thử tự động một cách cấu trúc và dễ quản lý. |
| **RQ3**: Hiệu quả so với không dùng POM được thể hiện như thế nào? | POM sẽ giúp giảm thời gian chỉnh sửa mã khi giao diện thay đổi, giảm lỗi phát sinh từ việc trùng lặp mã và tăng độ ổn định của các bài kiểm thử tự động. |

3.4. Triển khai kế hoạch kiểm thử

Để triển khai kế hoạch kiểm thử tự động theo mô hình Page Object Model (POM), quy trình thực hiện được xây dựng dựa trên các bước trọng tâm bao gồm phân tích yêu cầu, thiết kế kịch bản, thiết lập môi trường, thiết kế mã code và kiểm thử, báo cáo như hình sau:



Đầu tiên, **phân tích yêu cầu** đóng vai trò nền tảng, đảm bảo việc kiểm thử tập trung vào các chức năng quan trọng của ứng dụng. Giai đoạn này yêu cầu xác định rõ các yếu tố chính cần kiểm tra, như các trang web chính, thành phần giao diện quan trọng, và các thao tác người dùng phổ biến. Đồng thời, cần xem xét các yêu cầu phi chức năng như hiệu năng, tính tương thích trình duyệt và khả năng mở rộng của ứng dụng.

Sau khi nắm bắt yêu cầu, bước tiếp theo là **thiết kế kịch bản kiểm thử**, trong đó xác định các trường hợp kiểm thử (test cases) dựa trên các yêu cầu đã phân tích. Mỗi trường hợp kiểm thử cần được mô tả chi tiết về mục tiêu, điều kiện đầu vào, thao tác cần thực hiện và kết quả mong đợi. Ví dụ, một kịch bản kiểm thử có thể kiểm tra chức năng đăng nhập hợp lệ, trong đó người dùng nhập đúng thông tin đăng nhập và được chuyển hướng đến trang chủ.

Tiếp theo là **thiết lập môi trường kiểm thử**, bao gồm cài đặt các công cụ hỗ trợ và chuẩn bị hạ tầng cần thiết. Python được chọn làm ngôn ngữ lập trình kết hợp với thư viện Selenium để tự động hóa kiểm thử. Công cụ này yêu cầu cài đặt WebDriver tương ứng với trình duyệt, chẳng hạn ChromeDriver, để tương tác với giao diện người dùng. Môi trường phát triển cần được tổ chức theo cấu trúc thư mục rõ ràng, bao gồm các tệp dành riêng cho mô hình POM, các tệp kiểm thử, và các tiện ích hỗ trợ.

Bước quan trọng nhất là **thiết kế mã code kiểm thử**. Tại đây, mã kiểm thử được viết theo nguyên tắc tách biệt giữa logic của giao diện (Page Object) và kịch bản kiểm thử. Trong mô hình POM, mỗi trang trong ứng dụng được biểu diễn dưới dạng một lớp Python, với các phần tử giao diện và phương thức tương ứng để thao tác với giao diện. Các lớp này đóng vai trò như một lớp trừu tượng, giúp tăng tính tái sử dụng và giảm sự phụ thuộc khi giao diện thay đổi. Kịch bản kiểm thử sẽ sử dụng các lớp này để thực hiện các thao tác và kiểm tra kết quả.

Cuối cùng là giai đoạn **kiểm thử và báo cáo**. Các bài kiểm thử được thực thi trên môi trường đã thiết lập để xác minh rằng ứng dụng hoạt động đúng theo yêu cầu. Hiệu quả của POM được đánh giá qua việc đo lường thời gian sửa đổi khi giao diện thay đổi, khả năng tái sử dụng mã, và độ ổn định của các bài kiểm thử. Kết quả thực thi được tổng hợp và phân tích, sau đó được trình bày trong báo cáo cuối cùng. Báo cáo này không chỉ bao gồm các kết quả kiểm thử mà còn đánh giá hiệu quả của mô hình POM so với phương pháp kiểm thử truyền thống, cung cấp cơ sở dữ liệu để đưa ra quyết định cho các dự án trong tương lai

3.4.1. Phân tích yêu cầu các chức năng

Trang web blog có các chức năng chính gồm Đăng ký, Đăng nhập, Thêm danh mục, Sửa danh mục, Xóa danh mục, và Đăng bài viết, việc phân tích yêu cầu kiểm thử tập trung vào các luồng tương tác chính, các điều kiện đầu vào/đầu ra, cũng như các trường hợp ngoại lệ. Dưới đây là chi tiết phân tích cho từng chức năng:

1. Chức năng đăng ký:

Chức năng Đăng ký yêu cầu người dùng cung cấp thông tin cần thiết để tạo tài khoản. Các yêu cầu kiểm thử được xây dựng để đảm bảo rằng hệ thống xử lý đúng các trường hợp hợp lệ và không hợp lệ:

1. Kiểm tra các trường có tồn tại trên trang: Trang phải hiển thị đầy đủ các text box và button cho phép tương tác
2. **Trường bắt buộc không được để trống**: Hệ thống phải hiển thị thông báo lỗi nếu các trường thông tin như tên, email, mật khẩu, hoặc xác nhận mật khẩu bị bỏ trống.
3. **Định dạng email hợp lệ**: Email phải tuân theo định dạng chuẩn (vd: example@domain.com), và hệ thống hiển thị thông báo lỗi nếu email không hợp lệ.
4. **Mật khẩu khớp nhau**: Mật khẩu và xác nhận mật khẩu phải giống nhau; nếu không, hệ thống cần hiển thị thông báo lỗi.
5. **Độ mạnh của mật khẩu**: Mật khẩu phải có độ dài tối thiểu 6 ký tự. Nếu không đáp ứng, hệ thống phải thông báo lỗi.
6. **Đăng ký hợp lệ**: Khi cung cấp thông tin hợp lệ, tài khoản được tạo thành công, và người dùng được chuyển đến giao diện tài khoản hoặc màn hình đăng nhập thành công.

**Từ đó, ta có bảng các test cases như sau**:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Tên Test Case** | **Bước thực hiện** | **Kết quả mong đợi** |
| TC01 | Kiểm tra các phần tử xuất hiện hợp lệ | Truy cập trang đăng ký. Xác minh các phần tử: trường nhập Tên, Email, Mật khẩu, Xác nhận mật khẩu, nút Đăng ký xuất hiện đúng vị trí và trạng thái. | Tất cả các phần tử (input Tên, Email, Mật khẩu, Xác nhận mật khẩu, nút Đăng ký) xuất hiện đầy đủ, khả dụng, đúng định dạng. |
| TC02 | Kiểm tra bỏ trống tất cả trường | Để trống các trường: Tên, Email, Mật khẩu, Xác nhận mật khẩu. Nhấn nút Đăng ký. | Hiển thị thông báo lỗi: "Please add all fields". |
| TC03 | Email không hợp lệ | Nhập Tên: "John Doe", Email: "invalid-email", Mật khẩu: "Secure@123", Xác nhận mật khẩu: "Secure@123". Nhấn nút Đăng ký. | Hiển thị thông báo lỗi: "Email must be a valid email address". |
| TC04 | Mật khẩu không khớp | Nhập Tên: "John Doe", Email: "johndoe@example.com", Mật khẩu: "Secure@123", Xác nhận mật khẩu: "Different@123". Nhấn nút Đăng ký. | Hiển thị thông báo lỗi: "Mật khẩu không khớp". |
| TC05 | Mật khẩu quá yếu | Nhập Tên: "John Doe", Email: "johndoe@example.com", Mật khẩu: "123", Xác nhận mật khẩu: "123". Nhấn nút Đăng ký. | Hiển thị thông báo lỗi: "Password must be length of greater than 6". |
| TC06 | Đăng ký hợp lệ | Nhập Tên: "John Doe", Email: email ngẫu nhiên hợp lệ, Mật khẩu: "Secure@123", Xác nhận mật khẩu: "Secure@123". Nhấn nút Đăng ký. | Đăng ký thành công, chuyển đến màn hình tài khoản với nút tài khoản hiển thị trên giao diện. |

1. Chức năng đăng nhập

Chức năng Đăng nhập yêu cầu người dùng cung cấp thông tin xác thực hợp lệ (email và mật khẩu). Các yêu cầu kiểm thử chính bao gồm:

1. Xác minh hiển thị đầy đủ: Trang đăng nhập phải hiển thị đầy đủ các textbox và button cho phép người dùng tương tác
2. **Trường thông tin bắt buộc:** Hệ thống phải hiển thị thông báo lỗi nếu bất kỳ trường nào (email hoặc mật khẩu) bị bỏ trống.
3. **Người dùng không tồn tại:** Hệ thống phải hiển thị thông báo lỗi nếu thông tin người dùng không được tìm thấy trong cơ sở dữ liệu.
4. **Mật khẩu không đúng:** Hệ thống cần thông báo rõ ràng nếu mật khẩu không đúng.
5. **Đăng nhập thành công:** Khi cung cấp thông tin hợp lệ, hệ thống phải cho phép đăng nhập và chuyển hướng đến giao diện chính của người dùng.

Với bảng test case chức năng đăng nhập là:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Tên Test Case** | **Bước thực hiện** | **Kết quả mong đợi** |
| TC01 | Kiểm tra các phần tử xuất hiện hợp lệ | Truy cập trang đăng nhập. Xác minh các phần tử giao diện (trường Email, trường Mật khẩu, nút Đăng nhập) hiển thị đúng định dạng và trạng thái khả dụng. | Tất cả các phần tử xuất hiện đầy đủ, khả dụng, đúng định dạng. |
| TC02 | Kiểm tra bỏ trống cả hai trường | Để trống Email và Mật khẩu. Nhấn nút Đăng nhập. | Hiển thị thông báo lỗi: "Không tồn tại người dùng này". |
| TC03 | Kiểm tra bỏ trống trường Mật khẩu | Nhập Email: admin@gmail.com, để trống Mật khẩu. Nhấn nút Đăng nhập. | Hiển thị thông báo lỗi: "Sai mật khẩu". |
| TC04 | Đăng nhập với mật khẩu không đúng | Nhập Email: admin@gmail.com, Mật khẩu: admin1234. Nhấn nút Đăng nhập. | Hiển thị thông báo lỗi: "Sai mật khẩu". |
| TC05 | Đăng nhập thành công | Nhập Email: admin@gmail.com, Mật khẩu: admin123. Nhấn nút Đăng nhập. | Đăng nhập thành công, chuyển đến giao diện chính của người dùng, nút tài khoản hiển thị trên giao diện. |

1. Chức năng quản lý danh mục bài viết

Chức năng quản lý danh mục bài viết cho phép quản trị viên quản lý danh mục tại đây. Bao gồm các yêu cầu chính như sau:

1. **Thêm danh mục mới:** Hệ thống cần xử lý và hiển thị thông báo khi thêm danh mục trùng tên hoặc thêm danh mục thành công với tên mới.
2. **Chỉnh sửa danh mục:** Hệ thống cần hỗ trợ sửa tên danh mục hiện tại, đảm bảo tên mới cập nhật đúng và không còn tên cũ trong danh sách.
3. **Xóa danh mục:** Xóa danh mục thành công cần hiển thị thông báo và đảm bảo danh mục bị xóa không còn xuất hiện.

**Với bảng test cases sẽ là:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Tên Test Case** | **Bước thực hiện** | **Kết quả mong đợi** |
| TC01 | Thêm danh mục trùng tên | Nhập tên danh mục đã tồn tại, nhấn nút "Thêm danh mục". | Hiển thị thông báo: "Thêm danh mục thất bại: undefined". |
| TC02 | Thêm danh mục mới | Nhập tên danh mục không tồn tại, nhấn nút "Thêm danh mục". | Hiển thị thông báo: "Thêm danh mục thành công". |
| TC03 | Chỉnh sửa tên danh mục | Chọn danh mục bất kỳ, sửa tên danh mục sang tên mới chưa tồn tại, nhấn nút "Cập nhật". | Hiển thị thông báo: "Cập nhật danh mục thành công". Tên cũ không còn xuất hiện trong danh sách. |
| TC04 | Xóa danh mục | Chọn danh mục bất kỳ, nhấn nút "Xóa", xác nhận bằng nút "Xác nhận". | Hiển thị thông báo: "Xóa danh mục thành công". Tên danh mục không còn xuất hiện trong danh sách. |

1. Chức năng thêm bài viết mới

Chức năng thêm bài viết mới cho phép người dùng tạo bài viết với các thành phần như: tiêu đề, nội dung, hình ảnh, và lựa chọn danh mục. Sau khi người dùng nhập đủ thông tin và nhấn "Đăng bài", hệ thống sẽ kiểm tra các điều kiện cần thiết như yêu cầu điền đầy đủ thông tin và hiển thị thông báo thành công khi bài viết được đăng tải. Cũng cần đảm bảo các hình ảnh được tải lên đúng cách và bài viết hiển thị đúng sau khi đăng.

Các yêu cầu chính bao gồm:

1. Tải ảnh thành công: Chức năng cho phép người dùng tải lên một ảnh thành công và hiển thị nó làm ảnh bìa
2. Lỗi dữ liệu không hợp lệ: Cần thông báo lỗi cho người dùng khi bài viết đăng bài không đầy đủ các thông tin cần thiết. Dữ liệu bắt buộc có là ảnh bìa, tiêu đề bài viết, danh mục và nội dung
3. Kiểm tra các thông báo: Trang web cần hiển thị các thông báo cho người dùng trong trường hợp lỗi hoặc đã đăng bài thành công

Với các test case là:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Tên Test Case** | **Bước thực hiện** | **Kết quả mong đợi** |
| TC01 | Thông báo khi không điền đủ thông tin | Không điền đủ thông tin, nhấn nút "Đăng bài". | Hiển thị thông báo lỗi: "Tất cả các trường phải được điền". |
| TC02 | Tải lên hình ảnh thành công | Tải lên hình ảnh, điền tiêu đề, chọn danh mục, nhập nội dung và nhấn "Đăng bài". | Hiển thị thông báo thành công: "Bài viết đã được đăng thành công". |
| TC03 | Kiểm tra bài viết đăng với thông tin đầy đủ | Điền đầy đủ thông tin, tải lên hình ảnh, chọn danh mục và nhấn "Đăng bài". | Bài viết đăng thành công và thông báo "Bài viết đã được đăng thành công" xuất hiện. |

* + 1. Phân tích và lựa chọn môi trường kiểm thử

**Phân tích và cài đặt môi trường kiểm thử** là một bước quan trọng trong quá trình phát triển phần mềm, giúp đảm bảo rằng các tính năng và yêu cầu của phần mềm sẽ được kiểm tra một cách chính xác và hiệu quả. Điều này giúp quá trình kiểm thử diễn ra suôn sẻ và không gặp trục trặc.

Trước khi cài đặt môi trường kiểm thử, cần xác định rõ các yêu cầu và phạm vi kiểm thử của dự án. Những câu hỏi đặt ra cần trả lời được là:

* **Loại kiểm thử**: Xác định loại kiểm thử bạn muốn thực hiện (kiểm thử chức năng, kiểm thử hiệu suất, kiểm thử bảo mật, kiểm thử giao diện người dùng, kiểm thử API, v.v.).
* **Môi trường kiểm thử**: Quyết định môi trường kiểm thử (ví dụ: kiểm thử trên trình duyệt, ứng dụng di động, hệ thống backend, v.v.).
* **Công cụ kiểm thử**: Chọn công cụ kiểm thử phù hợp (Selenium cho kiểm thử ứng dụng web, PyTest hoặc unittest cho kiểm thử tự động, v.v.).
* **Ngôn ngữ lập trình**: Chọn ngôn ngữ lập trình để kết hợp với công cụ kiểm thử nhằm đảm bảo về hiệu suất
* **Cấu trúc dự án**: Phân chia cấu trúc dự án theo cấu trúc nào để tối đa hóa được hiệu suất làm việc
* **Tạo báo cáo tự động**: Sau khi thực thi các test case thành công, cần có báo cáo tự động được sinh ra để xem tổng quát các test case.

Để giải quyết những câu hỏi trên, các biện pháp và công cụ được đưa ra là:

* Loại kiểm thử: Kiểm thử trang web theo selenium là dạng kiểm thử chức năng kết hợp cùng với kiểm thử giao diện người dùng
* Ngôn ngữ lập trình: Python là một ngôn ngữ lập trình cấp cao, dễ học và sử dụng, được thiết kế với cú pháp đơn giản, rõ ràng và dễ đọc. Python hỗ trợ nhiều kiểu dữ liệu, thư viện phong phú và ứng dụng trong nhiều lĩnh vực như phát triển phần mềm, khoa học dữ liệu, học máy, kiểm thử tự động và phát triển web
* Unittest: Unittest là một thư viện tích hợp sẵn trong Python để viết và chạy các bài kiểm thử. Nó giúp kiểm tra các thành phần riêng biệt (unit) của một ứng dụng, đảm bảo rằng từng phần của hệ thống hoạt động chính xác. unittest hỗ trợ việc tổ chức các bài kiểm thử thành các **test case** và **test suite**, đồng thời cung cấp các phương thức để xác nhận các giá trị kỳ vọng trong quá trình kiểm thử.
* Selenium là một công cụ kiểm thử tự động mạnh mẽ dành cho ứng dụng web. Nó cung cấp các API để tương tác với trình duyệt và thực hiện các tác vụ như click, nhập liệu, cuộn trang, v.v. Selenium hỗ trợ nhiều trình duyệt và có thể được sử dụng với nhiều ngôn ngữ lập trình như Python, Java, C#, v.v. Công cụ này thường được dùng để kiểm thử chức năng và giao diện người dùng của ứng dụng web.
* Cấu trúc dự án: **Page Object Model (POM)** là một mẫu thiết kế phần mềm giúp tạo ra các lớp đối tượng đại diện cho các trang trong ứng dụng web. Mục tiêu chính của POM là tách biệt mã kiểm thử khỏi mã giao diện người dùng, giúp việc bảo trì và phát triển kiểm thử trở nên dễ dàng hơn. Các thư mục trong tập tin được chia thành 2 thư mục chính bao gồm: thư mục pages Chứa các lớp **Page Object** cho từng trang của ứng dụng, mỗi lớp đại diện cho một trang và chứa các phương thức để thao tác với các phần tử trên trang đó; thư mục test-cases chứa các tệp kiểm thử, mỗi tệp thường đại diện cho một nhóm chức năng của ứng dụng, các bài kiểm thử sử dụng các lớp trong thư mục pages/ để tương tác với giao diện người dùng và thực hiện các kiểm thử.
* Báo cáo tự động: **HTML Test Runner** là một công cụ giúp chạy các bài kiểm thử (tests) và hiển thị kết quả kiểm thử dưới dạng báo cáo HTML dễ đọc và trực quan. Thông thường, khi sử dụng các framework kiểm thử như unittest trong Python, kết quả sẽ được hiển thị trên terminal hoặc console, điều này có thể khó theo dõi nếu có quá nhiều bài kiểm thử hoặc nếu kết quả kiểm thử dài dòng. HTML Test Runner giúp giải quyết vấn đề này bằng cách tạo ra các báo cáo đẹp mắt, dễ hiểu và dễ chia sẻ dưới định dạng HTML.
  + 1. Cài đặt môi trường kiểm thử

Với các yêu cầu được đặt ra, môi trường kiểm thử cũng cần đảm bảo được sự linh hoạt và tính linh động nhằm thiết kế và chạy các đoạn mã một cách mượt mà, các công cụ sử dụng bao gồm:

* VSCode: Visual Studio Code (VSCode) là một trình soạn thảo mã nguồn mã nguồn mở, nhẹ và mạnh mẽ. Nó hỗ trợ nhiều ngôn ngữ lập trình như Python, JavaScript, C++, và có khả năng mở rộng qua các tiện ích mở rộng (extensions). VSCode cung cấp các tính năng như tô màu cú pháp, tự động hoàn thành mã, gỡ lỗi (debugging), kiểm thử, và tích hợp với Git. Nó được sử dụng phổ biến cho các dự án phần mềm và kiểm thử tự động.
* WebDriver: WebDriver là một công cụ giao tiếp giữa Selenium và các trình duyệt web để tự động hóa các tác vụ như kiểm thử. Nó cho phép Selenium điều khiển các trình duyệt (Google Chrome, Firefox, Safari, v.v.) để thực hiện các hành động như nhấp chuột, nhập liệu, lấy dữ liệu từ trang web, v.v. WebDriver cung cấp API cho các ngôn ngữ lập trình như Python, Java, C#, giúp tạo ra các kịch bản kiểm thử tự động. WebDriver cần được cấu hình đúng với trình duyệt tương ứng (ví dụ: ChromeDriver cho Google Chrome).

Bên cạnh đó, các thư viện cũng được cài đặt các phiên bản phù hợp với nhau để tranh xung đột. Bảng sau trình bày chi tiết phiên bản các thư viện:

|  |  |
| --- | --- |
| Tên thư viện | Phiên bản |
| Python | 3.10.0 |
| Selenium | 4.17.2 |
| Unittest | 3.10 |
| HtmlTestRunner | 1.2.1 |

* + 1. Xây dựng các đoạn mã
       1. Xây dựng các lớp Page

1. Chức năng đăng ký: Xây dựng class RegisterPage với phương thức và các tham số như sau:



Class RegisterPage là một lớp đại diện cho trang đăng ký trong mô hình **Page Object Model (POM)**. Nó cung cấp các phương thức để tương tác với các phần tử trên trang đăng ký như trường "Tên", "Email", "Mật khẩu", "Xác nhận mật khẩu" và nút "Đăng ký". Các phương thức trong class này bao gồm việc điền dữ liệu vào các trường nhập liệu và nhấn nút "Đăng ký". Mục đích là tách biệt logic giao diện người dùng với mã kiểm thử, giúp mã dễ duy trì và mở rộng.

1. Chức năng đăng nhập

Class LoginPage tương tự như RegisterPage, nhưng dùng để xử lý các hành động liên quan đến trang đăng nhập. Class được định nghĩa như sau:

A screenshot of a computer program

Description automatically generated

1. Chức năng quản lý danh mục

A screenshot of a computer program

Description automatically generated

1. Chức năng đăng bài

A screen shot of a computer program

Description automatically generated

1. Chức năng cập nhật

A computer screen shot of a program code

Description automatically generated

* + - 1. Xây dựng các lớp testcase

1. Test cases đăng ký

A screen shot of a computer program

Description automatically generated

1. **Test cases đăng nhập**

**A screenshot of a computer program

Description automatically generated**

1. **Test case quản lý danh mục: thêm, xóa, sửa**

**A screen shot of a computer screen

Description automatically generated**