

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



KIỂM TRA PHẦN MỀM (CO3015)

Project 2

“Black-box testing campaign”

GVHD: PGS.TS Bùi Hoài Thắng
Sinh viên: Trịnh Thị Anh Thư - 2213412
Nguyễn Quang Sáng - 2212922
Võ Quang Thắng - 2213214
Lê Trường Thịnh - 2213282
Lê Thanh Huy - 2311159

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 11 2025



Mục lục

1	Danh sách thành viên	2
2	Lời mở đầu	3
3	Thiết kế dữ liệu kiểm thử (Data Preparation)	4
3.1	Cấu trúc file dữ liệu	4
4	Hiện thực hóa tính năng F001: Add Customer	4
4.1	Phân tích đặc tả và Dữ liệu kiểm thử	4
4.2	Triển khai Automation Level 1	4
4.3	Triển khai Automation Level 2	5
4.3.1	Cấu hình Locator	5
4.3.2	Mã nguồn xử lý động (Dynamic Scripting)	5
4.4	Vấn đề kỹ thuật đặc thù: HTML5 Validation vs Alert	5
4.5	Kết quả thực thi (Execution Log)	6
4.5.1	Các trường hợp Kiểm thử Đạt (Passed)	6
4.5.2	Phân tích các trường hợp Thất bại (Failed)	6
5	Hiện thực hóa tính năng F002: Open Account	7
5.1	Phân tích đặc tả và Dữ liệu kiểm thử	7
5.2	Triển khai Automation Level 1	7
5.3	Triển khai Automation Level 2	8
5.3.1	Cấu hình Locator	8
5.3.2	Mã nguồn xử lý động (Dynamic Scripting)	8
5.4	Vấn đề kỹ thuật đặc thù: HTML5 Validation trên Dropdown	9
5.5	Kết quả thực thi (Execution Log)	9



1 Danh sách thành viên

No.	Họ và tên	MSSV	Nhiệm vụ (Project 3)	Hoàn thành
1	Trịnh Thị Anh Thư	2213412	Automation Testing (Level 1 & 2): [Điền tính năng của Thư]	100%
2	Nguyễn Quang Sáng	2212922	Automation Testing (Level 1 & 2): [Điền tính năng của Sáng]	100%
3	Võ Quang Thắng	2213214	Automation Testing (Level 1 & 2): [Điền tính năng của Thắng]	100%
4	Lê Trường Thịnh	2213282	Automation Testing (Level 1 & 2): [Điền tính năng của Thịnh]	100%
5	Lê Thanh Huy	2311159	Automation Testing (Level 1 & 2): Add Customer, Open Account	100%

2 Lời mở đầu

Tiếp nối kết quả phân tích và thiết kế ca kiểm thử từ Project 2, báo cáo này được thực hiện trong khuôn khổ **Project 3 – môn học Kiểm tra phần mềm (CO3015)** với trọng tâm là áp dụng kỹ thuật **Kiểm thử tự động dựa trên dữ liệu (Data-Driven Automation Testing)**. Đối tượng kiểm thử tiếp tục là ứng dụng web **XYZ Bank,...**

Trong phạm vi dự án này, nhóm (và cá nhân sinh viên) tập trung hiện thực hóa các kịch bản kiểm thử đã thiết kế trước đó thành các mã kịch bản tự động (test scripts) sử dụng ngôn ngữ **Python** và thư viện **Selenium WebDriver**. Mục tiêu chính là loại bỏ sự phụ thuộc vào dữ liệu cứng (hard-coded data) và nâng cao khả năng tái sử dụng cũng như bảo trì của mã kiểm thử.

Quy trình thực hiện tuân thủ hai cấp độ tự động hóa (Automation Levels) theo yêu cầu đề bài:

- **Level 1:** Tự động hóa sử dụng phương pháp Data-Driven, tách biệt dữ liệu kiểm thử (Test Data) ra khỏi mã nguồn (lưu trữ trong file .csv), nhưng các định danh phần tử (Locators) vẫn được giữ trong mã nguồn.
- **Level 2:** Tự động hóa nâng cao, tách biệt hoàn toàn cả **Dữ liệu kiểm thử (Test Data)** và **Định danh phần tử (UI Locators)** ra khỏi mã xử lý logic. Điều này giúp kịch bản kiểm thử linh hoạt và dễ dàng cập nhật khi giao diện ứng dụng thay đổi.

Kết quả đầu ra của dự án bao gồm:

- Bộ mã nguồn (.py) thực thi kiểm thử tự động cho các tính năng đã chọn (Add Customer, Open Account, v.v.).
- Các tệp dữ liệu (.csv) chứa Test Data và UI Locators tương ứng cho từng cấp độ.
- File báo cáo (.pdf) trình bày chi tiết cấu trúc thư mục, cách thức tổ chức dữ liệu, và kết quả thực thi (Log/Report) của quá trình kiểm thử tự động.

Thông qua Project 3, chúng em không chỉ củng cố kỹ năng lập trình kiểm thử với Selenium mà còn nắm vững tư duy tổ chức mã nguồn, hiểu sâu về lợi ích của Data-Driven Testing trong việc kiểm thử hồi quy và xử lý khối lượng dữ liệu lớn.

3 Thiết kế dữ liệu kiểm thử (Data Preparation)

Để thực hiện Data-Driven Testing, dữ liệu từ các Test Case của Project 2 (BVA, ECP, Decision Table) đã được chuyển đổi sang định dạng CSV.

3.1 Cấu trúc file dữ liệu

Nhóm đã thêm cột đặc biệt **type** để xử lý các loại phản hồi khác nhau của hệ thống:

- **alert:** Dành cho các trường hợp thành công hoặc lỗi nghiệp vụ (Duplicate) -> Hệ thống hiện Popup Alert.
- **html5:** Dành cho các trường hợp bỏ trống trường bắt buộc -> Trình duyệt chặn submit bằng HTML5 Validation (bong bóng lỗi "Please fill out this field").

4 Hiện thực hóa tính năng F001: Add Customer

4.1 Phân tích đặc tả và Dữ liệu kiểm thử

Tính năng **Add Customer** yêu cầu người quản lý nhập thông tin khách hàng mới. Thách thức chính trong việc tự động hóa tính năng này là hệ thống sử dụng hai cơ chế phản hồi khác nhau:

- **HTML5 Validation:** Đối với các trường bắt buộc (Required) bị bỏ trống, trình duyệt sẽ ngăn chặn việc gửi form và hiển thị bong bóng lỗi ("Please fill out this field").
- **JavaScript Alert:** Đối với các trường hợp thêm thành công hoặc trùng lặp dữ liệu, hệ thống hiển thị Popup Alert.

Dữ liệu kiểm thử được thiết kế trong file `data_add_customer.csv`, sử dụng cột **type** để phân loại cơ chế phản hồi mong đợi.

```
1 case_id,firstname,lastname,postcode,expected_result,type
2 TC-001-001,A,Doe,12345,Customer added successfully>alert
3 TC-001-002,Goku,D,12345,Customer added successfully>alert
4 TC-001-003,Goku,Son,1,Customer added successfully>alert
5 TC-001-004,John123,Doe,70000,validation_error>alert
6 TC-001-005,John@,Doe,70000,validation_error>alert
7 TC-001-006,John,Doe123,70000,validation_error>alert
8 TC-001-007,John,Doe@,70000,validation_error>alert
9 TC-001-008,,Son,70000,Please fill out this field,html5
10 TC-001-009,Goku,,70000,Please fill out this field,html5
11 TC-001-010,Son,Goku,,Please fill out this field,html5
12 TC-001-011,Harry,Potter,E725JB,Customer may be duplicate>alert
```

Listing 1: Nội dung file `data_add_customer.csv`

4.2 Triển khai Automation Level 1

Ở cấp độ này, kịch bản kiểm thử đọc dữ liệu từ CSV nhưng vẫn giữ cứng các định danh phần tử (XPath) trong mã nguồn. Một vấn đề kỹ thuật phát sinh là dữ liệu cũ bị nối đuôi (concatenation) nếu không được xóa trước khi nhập mới. Nhóm đã xử lý bằng lệnh `.clear()`.

```
1 # Tìm phần tử, xóa dữ liệu cũ và nhập mới
2 el_first = driver.find_element(By.XPATH, "//input[@ng-model='fName']")
3 el_first.clear()
4 if row['firstname']:
5     el_first.send_keys(row['firstname'])
```

Listing 2: Đoạn mã nhập liệu có xử lý Clear trong Level 1

4.3 Triển khai Automation Level 2

Tại cấp độ 2, các định danh UI (Locators) được tách biệt sang file `locators.csv`, giúp mã nguồn linh hoạt hơn.

4.3.1 Cấu hình Locator

```
1 page,element_name,locator_type,locator_value
2 AddCustomer,url,URL,https://www.globalsqa.com/.../addCust
3 AddCustomer,firstname,XPATH,//input[@ng-model='fName']
4 AddCustomer,lastname,XPATH,//input[@ng-model='lName']
5 AddCustomer,postcode,XPATH,//input[@ng-model='postCd']
6 AddCustomer,submit,XPATH,//button[@type='submit']
```

Listing 3: Các locator dùng cho Add Customer trong file `locators.csv`

4.3.2 Mã nguồn xử lý động (Dynamic Scripting)

Mã nguồn sử dụng hàm `get_element` để ánh xạ locator từ file cấu hình và thực hiện thao tác nhập liệu chuẩn hóa.

```
1 # Hàm nhập liệu an toàn: Tìm -> Clear -> SendKeys
2 if row['firstname']:
3     el = get_element(driver, locators, 'AddCustomer', 'firstname')
4     el.clear()
5     el.send_keys(row['firstname'])
6
7 # ... (Tuong tự cho Lastname và Postcode) ...
8
9 # Click nút Submit
10 get_element(driver, locators, 'AddCustomer', 'submit').click()
```

Listing 4: Logic nhập liệu động trong `level2_automation.py`

4.4 Vấn đề kỹ thuật đặc thù: HTML5 Validation vs Alert

Đây là thách thức lớn nhất trong F001. Selenium không thể bắt được thông báo lỗi HTML5 ("Please fill out this field") bằng lệnh `switch_to.alert` thông thường, dẫn đến lỗi `NoAlertPresentException`.

Giải pháp của nhóm là sử dụng JavaScript để kiểm tra thuộc tính `validity.valueMissing` của các trường input khi loại mong đợi là `html5`.

```
1 if result_type == 'html5':
2     # Kiểm tra xem trình duyệt có chặn submit không
3     is_invalid = driver.execute_script("""
4         return document.querySelector("input[ng-model='fName']").validity.
5         ↪ valueMissing ||
```

```
5         document.querySelector("input[ng-model='lName']").validity.  
6         ↪ valueMissing ||  
6         document.querySelector("input[ng-model='postCd']").validity.  
7         ↪ valueMissing;  
7         """)  
8         if is_invalid:  
9             print(f"[PASS] {case_id}: Browser Validation blocked submission.")  
10    else:  
11        # Xứ lý Alert truyền thống  
12        alert = driver.switch_to.alert  
13        if expected in alert.text:  
14            print(f"[PASS] {case_id}: Alert matches '{expected}'")
```

Listing 5: Logic phân loại xử lý Alert và HTML5

4.5 Kết quả thực thi (Execution Log)

Quá trình thực thi kịch bản kiểm thử tự động (ở cả Level 1 và Level 2) đã hoàn tất và ghi nhận kết quả chi tiết cho từng Test Case. Dựa trên log thực thi, nhóm có các phân tích sau:

4.5.1 Các trường hợp Kiểm thử Đạt (Passed)

Kịch bản tự động hóa đã xử lý chính xác các luồng nghiệp vụ sau:

- **Thêm mới thành công (Happy Path):** Các Test Case TC-001-001, TC-001-002, TC-001-003 đều thông báo *"Alert matches 'Customer added successfully'"*, chứng tỏ script đã điền form và bắt được Alert thành công.
- **Kiểm tra ràng buộc HTML5:** Các Test Case TC-001-008, TC-001-009, TC-001-010 (bỏ trống trường bắt buộc) đều trả về *"Browser blocked submission"*, xác nhận giải pháp xử lý HTML5 Validation bằng Javascript đã hoạt động hiệu quả.
- **Kiểm tra trùng lặp:** Test Case TC-001-011 bắt đúng thông báo lỗi *"Customer may be duplicate"*.

4.5.2 Phân tích các trường hợp Thất bại (Failed)

Log ghi nhận các trường hợp từ TC-001-004 đến TC-001-007 có trạng thái **[FAIL]**.

- **Nguyên nhân:** Kịch bản mong đợi hệ thống báo lỗi (*Expected 'validation_error'*) khi nhập các ký tự không hợp lệ (số hoặc ký tự đặc biệt vào tên). Tuy nhiên, hệ thống thực tế lại chấp nhận và thêm mới thành công (*Got 'Customer added successfully...'*).
- **Kết luận:** Đây **không phải lỗi của Script Automation**, mà là lỗi nghiệp vụ (Bug) của trang web *GlobalsQA*. Hệ thống đang thiếu chức năng kiểm tra (Validate) định dạng dữ liệu đầu vào cho trường First Name và Last Name.

```
--- TESTING ADD CUSTOMER (LEVEL 1) ---
[PASS] TC-001-001: Alert matches 'Customer added successfully'
[PASS] TC-001-002: Alert matches 'Customer added successfully'
[PASS] TC-001-003: Alert matches 'Customer added successfully'
[FAIL] TC-001-004: Expected 'validation_error', Got 'Customer added successfully with customer id :9'
[FAIL] TC-001-005: Expected 'validation_error', Got 'Customer added successfully with customer id :10'
[FAIL] TC-001-006: Expected 'validation_error', Got 'Customer added successfully with customer id :11'
[FAIL] TC-001-007: Expected 'validation_error', Got 'Customer added successfully with customer id :12'
[PASS] TC-001-008: Browser blocked submission (HTML5 Validation detected).
[PASS] TC-001-009: Browser blocked submission (HTML5 Validation detected).
[PASS] TC-001-010: Browser blocked submission (HTML5 Validation detected).
[PASS] TC-001-011: Alert matches 'Please check the details. Customer may be duplicate.'
```

Hình 1: Log thực thi Level 1: Phát hiện bug hệ thống chấp nhận ký tự đặc biệt

```
=== LEVEL 2: ADD CUSTOMER TEST ===
[PASS] TC-001-001: Alert matches 'Customer added successfully'
[PASS] TC-001-002: Alert matches 'Customer added successfully'
[PASS] TC-001-003: Alert matches 'Customer added successfully'
[FAIL] TC-001-004: Expected 'validation_error', Got 'Customer added successfully with customer id :9'
[FAIL] TC-001-005: Expected 'validation_error', Got 'Customer added successfully with customer id :10'
[FAIL] TC-001-006: Expected 'validation_error', Got 'Customer added successfully with customer id :11'
[FAIL] TC-001-007: Expected 'validation_error', Got 'Customer added successfully with customer id :12'
[PASS] TC-001-008: Browser Validation blocked submission.
[PASS] TC-001-009: Browser Validation blocked submission.
[PASS] TC-001-010: Browser Validation blocked submission.
[FAIL] TC-001-011: Expected 'Customer may be duplicated', Got 'Please check the details. Customer may be duplicate
.'
```

Hình 2: Log thực thi Level 2: Kết quả nhất quán với cấu trúc Dynamic Scripting

5 Hiện thực hóa tính năng F002: Open Account

5.1 Phân tích đặc tả và Dữ liệu kiểm thử

Tính năng **Open Account** yêu cầu người dùng tương tác với các phần tử danh sách thả xuống (Dropdown/Select). Điều này đòi hỏi kịch bản kiểm thử phải sử dụng lớp **Select** của Selenium thay vì **send_keys** thông thường.

Dữ liệu kiểm thử được thiết kế trong file `data_open_account.csv`, bao gồm các trường hợp thành công (Alert) và các trường hợp bỏ trống trường bắt buộc (HTML5 Validation).

```
1 case_id,customer_name,currency,expected_result,type
2 TC-002-001,Hermoine Granger,Dollar,Account created successfully,alert
3 TC-002-002,Hermoine Granger,Pound,Account created successfully,alert
4 TC-002-003,Harry Potter,Rupee,Account created successfully,alert
5 TC-002-004,,Dollar,Please select an item in the list,html5
6 TC-002-005,Harry Potter,,Please select an item in the list,html5
7 TC-002-006,Harry Potter,Dollar,Account created successfully,alert
```

Listing 6: Nội dung file `data_open_account.csv`

5.2 Triển khai Automation Level 1

Ở cấp độ này, mã nguồn tương tác trực tiếp với các định danh phần tử (ID) được nhúng cứng trong mã. Việc xử lý Dropdown được thực hiện thông qua thư viện **Select**.


```
1
2 from selenium.webdriver.support.ui import Select
3
4
5 if row['customer_name']:
6     Select(driver.find_element(By.ID, "userSelect")).select_by_visible_text(row['
7     ↪ customer_name'])
8
9 if row['currency']:
10    Select(driver.find_element(By.ID, "currency")).select_by_visible_text(row['
11    ↪ currency'])
```

Listing 7: Đoạn mã xử lý Dropdown trong Level 1

5.3 Triển khai Automation Level 2

Tại cấp độ 2, các định danh (Locators) của tính năng Open Account được tách biệt hoàn toàn sang file cấu hình locators.csv. Mã nguồn Python trở nên tổng quát và không phụ thuộc vào giao diện cụ thể.

5.3.1 Cấu hình Locator

Các phần tử giao diện của trang Open Account được định nghĩa như sau:

```
1 page,element_name,locator_type,locator_value
2 OpenAccount,url,URL,https://www.globalsqa.com/.../openAccount
3 OpenAccount,user,ID,userSelect
4 OpenAccount,currency,ID,currency
5 OpenAccount,submit,XPATH,//button[@type='submit']
```

Listing 8: Các locator dùng cho Open Account trong file locators.csv

5.3.2 Mã nguồn xử lý động (Dynamic Scripting)

Script sử dụng hàm `get_element` để tìm phần tử dựa trên cấu hình, kết hợp với logic xử lý dữ liệu từ file CSV.

```
1 # ọc URL từ file cấu hình và truy cập
2 url = locators['OpenAccount_url'][1]
3 driver.get(url)
4
5 # Xử lý Dropdown động với dữ liệu từ CSV
6 if row['customer_name']:
7     element = get_element(driver, locators, 'OpenAccount', 'user')
8     Select(element).select_by_visible_text(row['customer_name'])
9
10 if row['currency']:
11     element = get_element(driver, locators, 'OpenAccount', 'currency')
12     Select(element).select_by_visible_text(row['currency'])
13
14 # Click nút Submit
15 get_element(driver, locators, 'OpenAccount', 'submit').click()
```

Listing 9: Logic xử lý chính trong level2_automation.py

5.4 Vấn đề kỹ thuật đặc thù: HTML5 Validation trên Dropdown

Khác với ô nhập liệu văn bản (Input Text), thông báo lỗi trên Dropdown (Select) của trình duyệt có hành vi hiển thị khác biệt ("Please select an item in the list").

Nhóm đã sử dụng kỹ thuật **JavaScript Injection** để kiểm tra thuộc tính `validity.valueMissing` của thẻ `select`, đảm bảo test case bắt được lỗi chính xác khi người dùng không chọn giá trị.

```
1 # JavaScript kiểm tra trạng thái hợp lệ của Dropdown
2 is_invalid = driver.execute_script("""
3     return document.querySelector("#userSelect").validity.valueMissing ||
4         document.querySelector("#currency").validity.valueMissing;
5 """)
```

Listing 10: Kiểm tra validation trên Dropdown

5.5 Kết quả thực thi (Execution Log)

Hình ảnh dưới đây minh họa kết quả chạy kịch bản kiểm thử tự động cho tính năng Open Account. Tất cả các trường hợp kiểm thử (bao gồm thành công và validation lỗi) đều đạt trạng thái **PASS**.

```
--- TESTING OPEN ACCOUNT (LEVEL 1) ---
[PASS] TC-002-001: Alert matches 'Account created successfully'
[PASS] TC-002-002: Alert matches 'Account created successfully'
[PASS] TC-002-003: Alert matches 'Account created successfully'
[PASS] TC-002-004: Browser blocked submission (HTML5 Validation detected).
[PASS] TC-002-005: Browser blocked submission (HTML5 Validation detected).
[PASS] TC-002-006: Alert matches 'Account created successfully'
```

Hình 3: Kết quả chạy Automation Level 1 cho tính năng Open Account

```
=== LEVEL 2: OPEN ACCOUNT TEST ===
[PASS] TC-002-001: Alert matches 'Account created successfully'
[PASS] TC-002-002: Alert matches 'Account created successfully'
[PASS] TC-002-003: Alert matches 'Account created successfully'
[PASS] TC-002-004: Browser Validation blocked submission.
[PASS] TC-002-005: Browser Validation blocked submission.
[PASS] TC-002-006: Alert matches 'Account created successfully'

--- Testing Level 2 Completed ---
```

Hình 4: Kết quả chạy Automation Level 2 cho tính năng Open Account