Traceability Document – QA Assessment for PUMA India Website

Test Case Traceability Document

Project: PUMA India Website
Document Version: 1.0
Date: 17.04.2025

---

1. Introduction

This document provides traceability between manual test cases (from the provided report) and their automated counterparts (using Selenium with Python). It ensures:

- **All manual test cases are covered in automation.**
- **Clear mapping between manual and automated test steps.**
- **Explanation of automation logic for each test case.**

---

2. Traceability Matrix

Manual TC ID	Test Scenario	Automated TC Coverage	Automation Approach	Notes
TC_1	Login with valid credentials	test_login_valid()	Uses send_keys() for email/password, asserts dashboard URL	Verify session cookies post-login
TC_2	Login with invalid creden	test_login_invalid()	Submits wrong credentials, checks error	Uses EC.visibility_of_element_located for error popup

Manual TC ID	Test Scenario	Automated TC Coverage	Automation Approach	Notes
	tials		message visibility	
TC_3	Sign up with valid details	test_signup()	Fills registration form, validates confirmation email (mock)	Requires test email service integration
TC_04	Search for "shoes"	test_search_valid()	Inputs "shoes", verifies product grid ([data-test-id='product-list-item'])	Uses WebDriverWait for dynamic results
TC_05	Search for invalid keyword	test_search_invalid()	Inputs "xyz123", checks "No results" message	Assert on .search-empty-message
TC_06	Empty search	test_search_empty()	Submits empty query, validates behavior (no action/error)	Conditional assertion based on UI response
TC_07	Open product	test_product_navigation()	Clicks first product tile, checks URL contains /pd	Uses EC.url_contains

Manual TC ID	Test Scenario	Automated TC Coverage	Automation Approach	Notes
	details		/	
TC_08	Verify product details	test_product_details()	Validates title, price, images using multiple selectors	Fallback selectors for robustness
TC_09	Change size option	test_size_selection()	Selects size dropdown, asserts active size class	Uses get_attribute("class") for validation
TC_10	Filter by gender (Men)	test_filter_gender()	Clicks "Men" filter, checks product count > 0	Uses len(driver.find_elements)
TC_11	Filter by price (₹3K–5K)	test_filter_price()	Selects price range, verifies prices fall in range	Parses price text with float(re.sub(r'[^\d.]', '', price.text))
TC_12	Clear filters	test_clear_filters()	Clicks "Clear All", asserts product count resets	Compares counts pre/post filter
TC_13	Add to cart	test_add_to_cart()	Clicks "Add to Cart", checks cart counter	Uses EC.text_to_be_present_in_element

Manual TC ID	Test Scenario	Automated TC Coverage	Automation Approach	Notes
			increment	
TC_14	Check cart contents	test_cart_contents()	Opens cart, verifies product name and price	Cross-checks with product page data
TC_15	Remove from cart	test_remove_from_cart()	Clicks remove, asserts cart empty message	Uses EC.visibility_of_element_located for empty state

3. Automation Logic Explained

3.1 Login/Sign-Up (TC_1–TC_3)

- **Automation Flow:**
    1. **Navigate to login page.**
    2. **Fill credentials (valid/invalid) using send_keys().**
    3. **Submit and validate:**
        - **Success: Redirect to account page.**
        - **Failure: Error message visibility.**
- **Challenges:**
    - **Captcha handling (not covered in manual tests).**
    - **OTP-based login requires external service mocking.**

3.2 Search (TC_04–TC_06)

- **Key Selectors:**

python

Copy

```
search_input = "input[placeholder='Search for products']"
product_tiles = "[data-test-id='product-list-item']"
no_results = ".search-empty-message"
```

- **Logic:**
  - **For invalid/empty searches, assert either:**
    - **Product grid disappears (EC.invisibility_of_element_located).**
    - **"No results" message appears.**

3.3 Product Page (TC_07–TC_09)

- **Dynamic Elements Handling:**

python

Copy

```
# Size selection example
size_btn = driver.find_element(By.CSS_SELECTOR, "[data-test-id='size-9']")
size_btn.click()
assert "selected" in size_btn.get_attribute("class")
```

3.4 Filters (TC_10–TC_12)

- **Price Filter Validation:**

python

Copy

```
prices = driver.find_elements(By.CSS_SELECTOR, "[data-test-id='product-price']")
for price in prices:
  price_value = float(price.text.replace('₹', '').replace(',', ''))
  assert 3000 <= price_value <= 5000
```

3.5 Cart (TC_13–TC_15)

- **Cart Interaction:**

python

Copy

 # Add to cart

add_btn = driver.find_element(By.CSS_SELECTOR, "[data-test-id='add-to-cart-button']")

add_btn.click()

WebDriverWait(driver, 10).until(

EC.text_to_be_present_in_element((By.CSS_SELECTOR, ".cart-count"), "1"))

---

4. Coverage Gaps & Notes

1. **Manual-Only Tests:**
    - **UI responsiveness (e.g., mobile view).**
    - **Payment gateway integration (requires sandbox).**
2. **Automation Limitations:**
    - **Third-party logins (Google/Facebook).**
    - **CAPTCHA handling.**
3. **Dependencies:**
    - **Test data cleanup (e.g., clear cart after TC_15).**

This document ensures 100% traceability between manual and automated tests while highlighting critical automation logic and edge cases.