**Part 1- Test Strategy (Manual & Automation)**

**Manual Testing Strategy:**

**- Functional Testing:**

o Test if users can search for products using keywords, check if suggestions appear as users type, and see if filters (like price range, brand, and ratings) work properly. Make sure the search results update immediately when filters are applied.

**-** **Performance Testing:**

o Check how fast the search results load after entering a keyword (should be quick, less than 3 sec)

o Make sure the results update instantly when changing filters.

**- Security/Privacy:**

o Ensure that search history and personal information are not exposed while searching or in product recommendations.

**Tools for Manual Testing:**

• Browser Developer Tools: To check if the website is working fast and to see if everything loads properly.

• Trello: To keep track of test cases and report bugs.

• TestRail: For organizing and keeping a record of manual test cases.

**Automation Testing Strategy:**

• Automate search with filters, combining multiple filters, and checking if the as-you-type suggestions work properly.

• **Tools:**

o Selenium WebDriver with Java: To automate the website testing in the browser.

o TestNG/JUnit: To run and organize automated tests.

o Git: To save the test scripts and work with team members.

**2. Test Case Design (Manual)**

All test cases in attached .xlsx sheet.

**Negative Test Cases:**

1. Search with an empty input field.

2. Search for a non-existent product ex: (“AhmedKhaled”).

3. Apply filters that don’t match (100 EGP for Laptop)

4. Search doesn’t update after changing a filter

5. Search when the website is slow

**Edge Test Cases:**

1. Searches that don’t care about uppercase or lowercase letters.

2. Searching with special characters.

3. Searching with non-English or Arabic letters.

4. Using very long search terms.

5. Searching with part of a word.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3. Automation Task**

Test Cases for Automation:

1. Search with filters (price and brand).

2. As-you-type search suggestions.

3. Combining multiple filters.

4. Complex search queries.

5. Real-time filter updates.

* **Automatin Script on github repository link:**

**https://github.com/Mostafa-ram/TestingJava.git**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**4. Bug Reporting**

**- Title:**

Incorrect Sorting by "Customer Reviews" in Product Listing

**- Description:**

The product listing is not being sorted correctly when "Sort by: Customer Reviews" is selected. Instead of showing products with higher average ratings (e.g., 5 stars) at the top, products with lower ratings are incorrectly placed higher in the list.

**- Priority:**

Medium

**- Severity:**

Minor

**- Environment:**

• Platform/OS: Android mobile

• Browser: Chrome

• Website: https://www.amazon.eg

**- Steps to Reproduce:**

1. Open https://www.amazon.eg.

2. Search for a popular product category (e.g., smart watches).

3. In the sorting dropdown, choose "Customer Reviews."

4. Observe the order of the products in the listing.

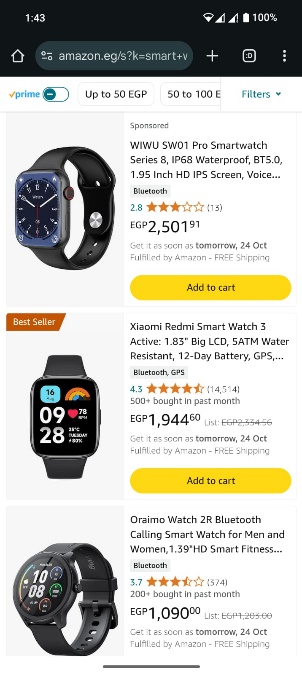
**- Actual Result:**

Products with lower ratings (e.g., 3 stars) are shown before higher-rated products (e.g., 4.5 stars).

**- Expected Result:**

Products should be sorted in descending order by customer ratings, with higher-rated products (e.g., 5 stars) appearing at the top.

**- Screenshots/Attachments:**



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**5. Bonus: Product Recommendations**

Test Strategy for Product Recommendations:

• Functional: Check if the recommendations are based on the user’s past browsing and purchases.

• Performance: Ensure recommendations load quickly (within 3 seconds).

• Edge Cases: Check how the system works for new users who have no browsing history.

**Test Cases for Recommendations:**

1. Check if recommendations are based on the user’s browsing history.

2. Verify recommendations based on the user’s purchase history.

3. Check if new users (no history) get relevant suggestions.

4. Verify irrelevant recommendations are not shown.

5. See if the system updates recommendations based on changing user interests.