1. Test Strategy

Overall Approach:
We will use a combination of automated end-to-end tests (focusing initially on core user journeys) and future incremental tests covering more functionalities. We will implement end-to-end tests using the Page Object Model (POM) in a Python-based Playwright framework.

Test Layers:

- **Smoke Test (Initial Focus):** Validate critical paths like the buy flow to ensure the application's fundamental features (browsing products and completing a purchase) are stable.
- **Functional Tests (Future Enhancement):**
    - **Log in/ Register functionality:** assert that users are able to register
    - **Sorting & Filtering:** Verify that sorting and filtering criteria work as expected and produce the correct product listing order.
    - **Reviews:** Confirm that users can submit and view product reviews.
    - **Wishlist:** Validate that items can be added to and displayed from a wishlist.
    - **[Explore more functionalities]**

Test Data:

- For the buy flow test, we will use generic valid customer information and a default product from the Bags category.
- For future sorting, filtering, and wishlist tests, we will select representative products and criteria.

Automation Tools and Frameworks:

- **Playwright with Python:** Fast and reliable browser automation.
- **Pytest:** For test execution and reporting.
- **Page Object Model (POM):** Ensures maintainable and scalable test code by encapsulating selectors and methods for each page.

Test Environments:

- **Test Environment:** `https://magento.softwaretestingboard.com/` (Production environment)
- **Browsers:** Start with Chrome (headless), and potentially scale to Firefox and Safari in the future.

Execution Frequency:

- **Buy Flow Test:**
    - Run on each new build as part of a smoke test suite to catch major regressions early.

- **Future Tests:**
    - Run daily or in scheduled CI pipeline runs for broader functional validation.

---

2. Rationale

Why This Buy Flow Test First?

- **Business Critical Path:** The ability for users to successfully place an order is the core business requirement. Without a functioning checkout, the store cannot meet its primary goal of generating revenue.
- **High Risk, High Value:** Any failure in the checkout process directly affects conversion rates and revenue. Testing it early and continuously is essential.
- **Foundational Test:** Validating the end-to-end flow ensures that all integrated components (catalog navigation, cart management, checkout, and order confirmation) work together. Once this foundation is stable, we can confidently expand into automating testing of more features.

Alignment with Use Cases and Business Logic:

- Customers must be able to find and purchase products smoothly. The test aligns with the end-user journey from browsing to completing a purchase.
- Other functionalities like sorting, filtering, and wishlist are important but not as critical as the ability to complete an actual transaction. These additional features enhance the user experience and help users find the right products, but they do not block revenue generation if temporarily malfunctioning.
- Prioritizing the buy flow aligns with business priorities: no successful orders mean no revenue, making this path the top testing priority.

Scalability and Maintenance Considerations:

- By implementing a POM structure and starting with critical tests first, we ensure a maintainable codebase that can easily scale as we add new test cases for sorting, filtering, reviews, and wishlist functionalities.
- When changes occur in the application's UI, only page objects need updating, minimizing effort and ensuring the tests remain robust and maintainable.

3. Test Case Description

Primary Test Case: Buy Flow (End-to-End)
Goal: Validate that a user can successfully browse products, add a product to the cart, proceed through the checkout, and place an order, ultimately reaching the “Thank You” page.

Preconditions:

- User is on the landing page of the website (`https://magento.softwaretestingboard.com/`).

Test Steps:

1. **Navigate to Gear Category:**
    - From the landing page, locate and click on the “Gear” top-level menu item.
    - In the “Gear” dropdown, select “Bags” to view the Bags product listing page.
2. **Change to Second Page of Results:**
    - On the Bags listing page, navigate to the second page of products. (Assume pagination controls are available and functional.)
3. **Select the First Item on Second Page:**
    - Identify the first product on the second page and click on it to open its Product Detail Page (PDP).
4. **Add to Cart:**
    - On the PDP, select any required product options (e.g., color or size) if applicable.
    - Click the “Add to Cart” button.
5. **Proceed to Checkout:**
    - Navigate to the cart by clicking the cart icon or cart link.
    - In the cart, click “Proceed to Checkout.”
6. **Fill in Checkout Form:**
    - Enter valid shipping and billing information (name, address, payment details, etc.).
7. **Place Order:**
    - Click “Place Order” or the equivalent final checkout button.
8. **Verify Thank You Page:**
    - Assert that the user is redirected to a “Thank You” or confirmation page, indicating a successful order placement.
    - Optionally, verify that the correct order number or confirmation message is displayed.

Expected Result:
The user completes the entire purchase flow without errors, and a thank you/confirmation page is displayed.