Test Case ID	Title	Description	Precondition	Priority	Steps	Expected Result	Actual Result	Conclusion
TC_AU_001	Register a new user and verify session persistence	This test verifies that a newly registered user remains logged in when the session is reused using cy. session().	No user is logged in. The user data is dynamically generated.	High	1. Navigate to the home page. 2. Register a new user with random data. 3. Visit the homepage again. 4. Verify that the user remains logged in. 5. Delete the account.	User is registered, session is persisted, and user remains logged in across reloads.	As expected	Pass
TC_AU_002	Login, logout, and login again with valid credentials	This test verifies the login and logout flow with the same user credentials.	A user account exists (created via API).	High	1. Log in with valid credentials. 2. Verify successful login. 3. Log out. 4. Verify redirection to homepage and login button visibility. 5. Log in again. 6. Verify login. 7. Delete the account.	User can log in, log out, and log in again successfully.	As expected	Pass
TC_AU_003	Register with an existing email address	This test verifies that the application does not allow registering an already registered email.	User already exists (created via API); user info available in userInfo fixture.	Medium	1. Navigate to the registration page. 2. Enter name and already registered email. 3. Submit the form.	An error message is displayed: "Email Address already exist!"	As expected	Pass
TC_AU_004	Attempt to login with incorrect password	Verifies that login fails with an incorrect password, even if the email is valid.	User account exists (created via API); correct credentials are known.	Medium	1. Go to the login page. 2. Enter valid email. 3. Enter incorrect password. 4. Submit login form.	Error message appears: "Your email or password is incorrect!"	As expected	Pass
TC_PO_005	Filter products by category and verify product details	Verifies that filtering by "Women > Dress" shows relevant products and validates details of a specific product (ID 3).	User must be on the "All Products" page (/products).	High	1. Click Women > Dress category link. 2. Assert that listed products include the keyword “Dress”. 3. Locate product with ID 3 and capture its name and price. 4. Click to open product detail page. 5. Verify the product detail page displays correct name, category, and price.	Product list is filtered correctly. Product detail page shows expected name, price, and category.	As expected	Pass
TC_PO_006	Search for a product by keyword and validate search results	Verifies that searching for a keyword (e.g., "Blue") returns appropriate product (s).	User must be on the "All Products" page.	Medium	1. Enter search term “Blue” in the search input. 2. Click the search button. 3. Verify that the result includes product titled “Blue Top”. 4. Confirm at least one visible product in result list.	Search result includes products matching the keyword “Blue”, including the expected "Blue Top".	As expected	Pass

TC_PO_007	Visiting invalid product ID should not display product information	Ensures visiting a non-existent product detail page shows no product information or cart options.	None – user does not need to be logged in; visiting a random invalid product ID directly.	Medium	1. Visit product detail page with a non-existent ID (/product_details/999999). 2. Verify that no .product-information or #quantity elements are present.	Product information section and quantity/cart options should not be visible.	The product detail page will have certain elements such as add to cart and quanity which should not be possible as the product does not exist. [Bug ID: BUG_PO_007]	Fail
TC_ORDER_001	Proceed to checkout, then register and complete the order	Ensures a guest can proceed to checkout, then register during the flow, and complete the order successfully.	Cart must contain at least one product before proceeding to checkout.	High	1. Add product ID 1 with quantity 1 to cart. 2. Click Cart and proceed to checkout. 3. Click Register / Login from the modal. 4. Complete registration with dynamic user info. 5. Return to cart and re-initiate checkout. 6. Verify delivery address matches registered info. 7. Add comment and place the order.	User should be able to register mid-checkout, place an order, and see the order success message.	As expected	Pass
TC_ORDER_002	Register first, then proceed to checkout and place an order	Validates the scenario where a user registers first, then adds product to cart and places an order successfully.	User must not exist in the system before test begins.	High	1. Go to login/signup page. 2. Register with random user data. 3. Add product to cart. 4. Go to cart and proceed to checkout. 5. Verify delivery address. 6. Add comment and place order. 7. Enter payment details and submit. 8. Confirm success message and continue.	User is able to register, complete checkout, and see successful order confirmation.	As expected	Pass
TC_ORDER_003	Login with existing user, add product to cart and complete checkout	Confirms that an existing user can log in, add product to cart, and complete an order successfully.	User account should be created via API before test execution.	High	1. Create user using backend API. 2. Navigate to login page and log in. 3. Verify successful login. 4. Add product to cart. 5. Go to cart and proceed to checkout. 6. Verify delivery address. 7. Add comment and place order. 8. Enter payment details and submit.	Logged-in user should be able to checkout and complete payment successfully.	As expected	Pass
TC_CART_001	Add multiple products to cart from different categories	Validate that products from different categories can be added to the cart and are visible.	None	Medium	1. Add product ID 1 to cart. 2. Add product ID 3 to cart. 3. Navigate to the Cart page.	Both product 1 and product 3 should be listed in the cart.	As expected	Pass
TC_CART_002	Change quantity of product in cart and verify update	Ensure the quantity of a product can be updated before adding it to the cart and reflected correctly in cart page.	Product should be in cart.	Medium	1. Go to product ID 3 detail page. 2. Set quantity to 2. 3. Click "Add to cart" and close modal. 4. Navigate to cart. 5. Verify quantity shows as 3 (previous + current).	Product 3 should display updated quantity in cart.	As expected	Pass

TC_CART_003	Validate cart total is calculated correctly	Confirms that unit price × quantity equals total price for each item in cart.	Products should be in cart with correct quantity.	High	1. Navigate to Cart. 2. Retrieve unit price and total price for product 1 (quantity = 1). 3. Retrieve unit price and total price for product 3 (quantity = 3). 4. Verify that total equals unit × quantity.	Total for each item should match expected calculation.	As expected	Pass
TC_CART_004	Remove a product from cart and ensure update reflects correctly	Ensure a user can remove an item from the cart and the cart reflects the updated list.	Product 1 and product 3 must be in cart.	Medium	1. Go to Cart. 2. Remove product ID 1 from cart. 3. Verify product 1 is no longer in the cart. 4. Confirm product 3 still exists.	Product 1 should not appear, product 3 should still be present.	As expected	Pass

Bug ID	Test Case ID	Title	Steps to Reproduce	Date Found	Status	Date Closed	Found By	Fixed By	Severity	Priority
BUG_PO_007	TC_PO_OO7	Invalid Product Detail Page Still Displays Product Information	1. Navigate to /product_details/999999 (or any clearly non-existent product ID). 2. Observe that the .product-information and #quantity elements are still rendered.	2025-05-26	Open		Rebof Katwal		Low	P3 (LOW)