👤 **Tester:** Paula Granado

🎯 **Test Objective:**
Validate core user flows on the Swag Labs platform, focusing on login, product browsing, sorting, filtering, cart management, and the complete checkout process. Ensure usability, accessibility, session handling, and system behavior are consistent across standard and edge-case user profiles.

🔒 **Prerequisites:**
- User must be logged in (except for Login tests)
- Tester must have access to:
  - Login Page: https://www.saucedemo.com/
  - Products Page: https://www.saucedemo.com/inventory.html
  - Cart Page: https://www.saucedemo.com/cart.html
  - Checkout Pages:
    - Step One: https://www.saucedemo.com/checkout-step-one.html
    - Step Two: https://www.saucedemo.com/checkout-step-two.html
    - Complete: https://www.saucedemo.com/checkout-complete.html

👥 **Test User Profiles:**
- standard_user – Valid user with normal behavior
- locked_out_user – Restricted user, cannot log in

🧪 **Test Credentials:**
Username: standard_user
Password: secret_sauce

🌐 **Test Environment:**
- Production demo site – https://www.saucedemo.com
- Tests executed in Google Chrome (v124), resolution 1920x1080

🛠 **Tools Used:**
- Google Sheets (Test Case Management)
- Google Docs (Reporting)
- Chrome DevTools (Basic inspection)

🔐 **Login Functionality:**
Validate login functionality with valid, invalid, and locked-out users. Ensure proper handling of errors, required fields, and credential feedback.

**Prerequisites:**
- Tester must have access to the SauceDemo login page: https://www.saucedemo.com/
- Test credentials must be available (e.g., standard_user, locked_out_user)
- Browser cache/cookies should be cleared to avoid session persistence

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Functional | TC01 | Valid Login with Correct Credentials | Verify the full login flow with valid credentials, ensuring no errors or delays. | The User is on the login page. | 1. Enter valid username.<br>2. Enter valid password.<br>3. Click Login. | Given the user is on the login page<br>When valid credentials are entered<br>And login is clicked<br>Then the user is redirected to the products page<br>And the "Products" title is displayed<br>And the product list is visible | User is immediately redirected to the products page without errors or delay. | User was redirected to the products page without errors or delay. Login flow worked as expected. | Passed | High | Critical |
| Negative | TC02 | Login Validation with invalid Username | Verify system behavior when an incorrect username is used. | The User is on the login page. | 1. Enter invalid username.<br>2. Enter valid password.<br>3. Click Login. | Given the user is on the login page<br>When an invalid username is entered<br>And the login button is clicked<br>Then an error message is displayed | Displays: “Epic sadface: Username and password do not match any user in this service.” | Displays: “Epic sadface: Username and password do not match any user in this service.” | Passed | High | Critical |
| Negative | TC03 | Login Validation with Invalid Password | Verify system behavior when an incorrect password is used. | The User is on the login page. | 1. Enter valid username.<br>2. Enter invalid password.<br>3. Click Login. | Given the user is on the login page<br>When an incorrect password is entered<br>And the login button is clicked<br>Then an error message is displayed | Displays: “Epic sadface: Username and password do not match any user in this service.” | Displays: “Epic sadface: Username and password do not match any user in this service.” | Passed | High | Critical |
| Negative | TC04 | Login Validation with Empty Fields | Ensure proper error message when both fields are left empty. | The User is on the login page. | 1. Leave username blank.<br>2. Leave password blank.<br>3. Click Login. | Given the user is on the login page<br>When the login button is clicked without entering credentials<br>Then an error message is displayed | Displays: “Epic sadface: Username and password are required.” | Displays only: “Epic sadface: Username is required.” | Failed | High | Critical |
| Negative | TC05 | Login Validation with Empty Username only | Ensure an error message is shown when username is missing. | The User is on the login page. | 1. Leave username blank.<br>2. Enter valid password.<br>3. Click Login. | Given the user is on the login page<br>When only a valid password is entered<br>And the login button is clicked<br>Then an error message is displayed | Displays: “Epic sadface: Username is required.” | Displays: “Epic sadface: Username is required.” | Passed | High | Critical |
| Negative | TC06 | Login Validation with Empty Password only | Ensure an error message is shown when password is missing. | The User is on the login page. | 1. Enter valid username.<br>2. Leave password blank.<br>3. Click Login. | Given the user is on the login page<br>When only a valid username is entered<br>And the login button is clicked<br>Then an error message is displayed | Displays: “Epic sadface: Password is required.” | Displays: “Epic sadface: Password is required.” | Passed | High | Critical |
| Security / Network | TC07 | Login Validation with Locked Out User | Ensure that users with a locked status cannot log in. | The User is on the login page. | 1. Enter username locked_out_user.<br>2. Enter valid password.<br>3. Click Login. | Given the user is a locked-out user<br>When the user enters valid credentials<br>And clicks the login button<br>Then an account locked error message is displayed | Displays: “Epic sadface: Sorry, this user has been locked out.” | Displays: “Epic sadface: Sorry, this user has been locked out.” | Passed | High | Critical |
| Security / Network | TC08 | Login Validation with SQL Injection Attempt | Ensure login is secure from basic SQL injection attempts. | The User is on the login page. | 1. Enter admin' OR '1'='1.<br>2. Enter any password.<br>3. Click Login. | Given the user attempts an SQL injection<br>When a malicious username is entered<br>Then the login should fail securely without exposing sensitive information | Login fails; system shows generic error and prevents injection. | Login fails; system shows generic error and prevents injection. | Passed | Medium | Critical |
| Usability / Responsiveness | TC09 | Login Validation with Password Masking | Ensure password input is masked (hidden). | The User is on the login page. | 1.Click on the password input field to focus it.<br>2. Type a sample password (e.g., “secret123”).<br>3. Observe that the characters typed are masked (displayed as asterisks or dots). | Given the user focuses on the password field<br>When the user types characters<br>Then the input should be masked using asterisks or dots | Password characters are shown as asterisks or dots. | Password characters are shown as asterisks or dots. | Passed | Medium | Minor |
| Usability / Responsiveness | TC10 | Responsive Login Page | Verify that the login form adjusts for mobile screens. | The User is on the login page. | 1. Open the login page.<br>2. Resize the browser to mobile screen size.<br>3. Verify the login form adapts and remains readable. | Given the user accesses the login page<br>When resize the browser to mobile screen size.<br>Then the login form should adapt responsively and remain readable. | Layout adapts properly when resized to mobile view. | Layout adapts properly when resized to mobile view. | Passed | Medium | Major |
| Usability / Responsiveness | TC11 | Tab Navigation Accessibility | Ensure keyboard (Tab key) navigation works properly. | The User is on the login page. | 1. Use the Tab key to navigate through all input fields and buttons.<br>2. Verify the focus moves logically and sequentially through interactive elements. | Given the user navigates the login form using the Tab key<br>Then the focus should move logically through all interactive fields | Tab navigation follows a logical, accessible order. | Tab navigation follows a logical, accessible order. | Passed | Medium | Major |
| Functional | TC12 | Login Validation with Caps Lock On | Ensure login fails when Caps Lock interferes. | The User is on the login page. | 1. Turn on Caps Lock.<br>2. Enter username and password with Caps Lock on.<br>3. Click Login. | Given the Caps Lock key is active<br>When the user enters credentials with Caps Lock on<br>Then an error message should be displayed if the credentials do not match | The system should show an error message indicating that the credentials are incorrect due to Caps Lock being on. | Login fails (as expected), but no Caps Lock warning shown. | Passed | Medium | Major |
| Functional | TC13 | Login Validation with Unlocked Previously Locked User | Verify that a previously locked user can log in after being unlocked. | The User is on the login page. | 1. Log in with a user that was locked.<br>2. Ensure the user is unlocked.<br>3. Attempt to log in again. | Given the user account is unlocked<br>When valid credentials are entered<br>Then the user is redirected to the products page | Login should be successful, and the user is redirected to the products page. | Blocked – backend access needed to unlock user. | Blocked | Medium | Major |
| Functional | TC14 | Login Validation with Multiple Clicks on Login Button | Ensure multiple clicks on login button don't cause errors or multiple requests. | The User is on the login page. | 1. Enter username and password.<br>2. Click the login button multiple times quickly (3-4 times). | Given the login button is clicked multiple times<br>When credentials are submitted<br>Then the login process is executed only once | Only one request should be processed, and the user should not encounter errors due to multiple clicks. | Only one request is processed, no issues observed. | Passed | Medium | Major |

🔐 **Login Functionality:**
Validate login functionality with valid, invalid, and locked-out users. Ensure proper handling of errors, required fields, and credential feedback.

**Prerequisites:**
- Tester must have access to the SauceDemo login page: https://www.saucedemo.com/
- Test credentials must be available (e.g., standard_user, locked_out_user)
- Browser cache/cookies should be cleared to avoid session persistence

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Navigation Post-login | TC15 | Navigation from Menu: About and Logout | Ensure that the About and Logout links function correctly after login. | The User is logged in and on the product page. | 1. On the product page, click on the burger menu (top-left corner).<br>2. Click on About or Logout link.<br>3. Check if the page redirects correctly. | Given the user is logged in and on the product page<br>When the About or Logout link is clicked<br>Then the user is redirected to the appropriate page (About page or login page) | • About → opens saucelabs.com.<br>• Logout → returns to login and clears session. | Clicking About redirected to saucelabs.com, and Logout redirected to the login page while ending the user session. | Passed | Medium | Critical |
| Negative | TC16 | Login While Offline | Ensure system reacts properly when attempting login without internet. | The User is on the login page. | 1. Turn on offline mode in the browser or disconnect the internet<br>2. Attempt to log in with valid credentials. | Given the user is offline<br>When a login attempt is made<br>Then a connection error message is displayed | Browser shows connection error; login fails as expected. | Browser displayed a connection error, and login failed due to lack of internet connectivity. | Passed | Medium | Critical |
| Functional | TC17 | Login Across Time Zones | Ensure login functionality is unaffected by system timezone. | The User is on the login page. | 1. Change the time zone of the computer or browser to a different one.<br>2. Attempt to log in with valid credentials. | Given the user's time zone is changed<br>When a login attempt is made<br>Then login succeeds without issues | Login succeeds regardless of timezone. | Login succeeded successfully despite changes in system time zone. | Passed | Low | Minor |
| Security / Network | TC18 | Login Validation using VPN or Proxy | Verify that login works or is properly blocked via VPN or proxy. | The User is on the login page. | 1. Enable VPN or proxy to change the user's IP address.<br>2. Attempt to log in with valid credentials. | Given the user is accessing via VPN or proxy<br>When a login attempt is made<br>Then the user is either logged in successfully or receives a message if geographical restrictions apply | Login should be processed successfully, or if restricted, an error message should be shown. | Login succeeds; no geo-blocking observed. | Passed | Medium | Minor |
| Functional | TC19 | Login Validation leading/trailing Spaces or Special Characters | Check if the system handles extra spaces or special characters. | The User is on the login page. | 1. Enter username with leading or trailing spaces (e.g., " standard_user ").<br>2. Enter password with special characters or extra spaces.<br>3. Attempt to log in. | Given the login fields contain spaces or special characters<br>When a login attempt is made<br>Then the system trims spaces or displays an error for invalid input | The system should handle extra spaces or special characters correctly by either trimming them or showing an error for invalid input. | System trimmed leading/trailing spaces or displayed an error for invalid characters, handling input correctly. | Passed | Medium | Major |
| Security | TC20 | Inactivity Session Expiration | Verify that user session expires after inactivity. | The User is on the login page. | 1. Login to the system.<br>2. Remain inactive for the timeout period. | Given the user is logged in<br>When the user remains inactive for the timeout period<br>Then the session expires and the user is redirected to the login page | User is automatically logged out after timeout period. | Session did not expire after long inactivity. | Failed | High | Critical |
| Security / Session | TC21 | Clear Session or Cookies | Ensure clearing browser session data forces logout. | The User is logged in on the products page. | 1. Login with valid credentials.<br>2. Open browser dev tools → Application tab.<br>3. Clear session storage and cookies.<br>4. Refresh the page. | Given the user is logged in<br>When session storage is cleared and the page is refreshed<br>Then the user is redirected to the login page or shown an error message | The session should end, and the user should be redirected to the login page or be denied access to protected resources. | Session storage and cookies were cleared successfully, causing the user to be redirected to the login page and access to protected resources was denied. | Passed | High | Critical |
| Functional | TC22 | Logout Flow and Redirection | Confirm that logging out ends the session and redirects to login page. | The User is on the login page. | 1. Login with valid credentials.<br>2. Click the burger menu.<br>3. Click "Logout".<br>4. Observe redirection. | Given the user is logged in<br>When the Logout link is clicked<br>Then the user is redirected to the login page and the session ends | Logout ends session and redirects to login page. | User session ended successfully, and the user was redirected to the login page after logout. | Passed | High | Critical |
| Boundary / Negative | TC23 | Character Limit | Check the maximum allowed characters in username and password fields. | The User is on the login page. | 1. Enter username with many characters (e.g., 256).<br>2. Enter password with many characters (e.g., 256).<br>3. Attempt login. | Given the user is on the login page<br>When a username and password with 256 characters are entered<br>And a login attempt is made<br>Then the login fails and a clear error message is displayed | Login fails or fields are rejected with a clear error message:<br>"Epic sadface: Username and password do not match any user in this service". | Login failed with a clear error message: "Epic sadface: Username and password do not match any user in this service," when using 256-character inputs. | Passed | Medium | Critical |
| Functional | TC24 | Login Validation with Special Characters / Unicode / Emoji | Ensure login fails appropriately when using invalid characters. | The User is on the login page. | 1. Enter username with special characters or Unicode (e.g., emojis).<br>2. Enter password similarly.<br>3. Attempt login. | Given the user is on the login page<br>When a username and password with special or Unicode characters are entered<br>And a login attempt is made<br>Then a login failure or clear error message is displayed | The user sees a clear error message:<br>"Epic sadface: Username and password do not match any user in this service". | User received a clear error message: "Epic sadface: Username and password do not match any user in this service." | Passed | Medium | Major |
| Functional / Security | TC25 | Manual Logout and Protected Page Access | Ensure that logging out invalidates session and prevents unauthorized access. | The User is on the login page. | 1. Login with a valid user.<br>2. Click Logout.<br>3. Attempt to access a protected page. | Given the user is logged in<br>When logout is clicked and a protected page is accessed<br>Then the session ends and access is denied | User is logged out, protected page access is denied. | User was successfully logged out, and access to protected pages was blocked until a new login. | Passed | High | Critical |
| Cross-browser / Usability | TC26 | Cross-browser Compatibility | Verify login functionality across Chrome, Firefox, Edge, Safari. | The User is on the login page. | 1. Open the login page on Chrome, Firefox, Edge, Safari.<br>2. Attempt login with valid user. | Given the user is on the login page<br>When the page is opened on Chrome, Firefox, Edge, and Safari<br>And login is attempted with valid credentials<br>Then login and navigation succeed on all browsers | Login works properly across all major browsers. | Login succeeded and the user was redirected to the inventory page without errors or visual issues across all tested browsers. | Passed | High | Critical |

📜 **Products Page:**
Test product page elements, including browsing, sorting, and filtering. Validate UI consistency and correct item display for all user profiles.

**Prerequisites:**
- User must be logged in as a valid user (e.g., standard_user)
- Access to the Products page: https://www.saucedemo.com/inventory.html
- At least one product must be available on the page for interaction

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Navigation | TC30 | Verify Navigation Links and Buttons | Verify that the links and buttons on the page redirect to the correct page. | The User is logged in and on the products page. | 1. On the product page, click on the burger menu (top-left corner).<br>2. Click on About or Logout link.<br>3. Check if the page redirects correctly. | Given the user is logged in and on the product page<br>When the About or Logout link is clicked<br>Then the user is redirected to the appropriate page (About page or login page) | The user should be redirected to the correct page when clicking on the links (e.g., About or Logout). | Navigation worked as expected for both links. | Passed | Medium | Major |
| Functional | TC31 | Verify Product Page Navigation and Interaction | Validate user’s ability to navigate to product details and add products to the cart on the products page. | The User is logged in and on the products page. | 1. On the product page, select a product to view its details.<br>2. Click ‘Add to cart’.<br>3. Verify the cart icon updates with the new item count. | Given the user is logged in and on the products page<br>When a product is selected and ‘Add to cart’ is clicked<br>Then the cart icon increments<br>And the item appears in the cart | Cart icon count increments and the cart contents include the newly added product after the user adds it from the product details. | Cart updated immediately and showed the correct product without issues. | Passed | Medium | Major |
| Functional | TC32 | Product Sorting | Verify that the product listing updates and displays items sorted according to the selected filter option (e.g., Price: low to high). | The User is logged in and on the products page. | 1. Navigate to the products page logged in.<br>2. Select the "Price low to high" sorting filter.<br>3. Confirm products reorder from lowest to highest price. | Given the user is on the products page<br>When the "Price low to high" filter is selected<br>Then the products are reordered from lowest to highest price | Products display in the correct sorted order without errors. | Products are correctly reordered by ascending price, displaying consistently without any sorting errors or display issues. | Passed | Medium | Minor |
| Functional | TC33 | View Product Details | Verify that the user can access and view detailed information of a product by clicking on its name or image. | The User is logged in and on the products page. | 1. On the product page, click on the name or image of any product.<br>2. Confirm the product details page displays accurate information. | Given the user is logged in on the products page<br>When the user clicks a product’s name or image<br>Then the product details page displays accurate name, description, and price | Product details page loads successfully and displays correct product name, description, image, and price. | As expected – the product details are displayed accurately upon click. | Passed | Medium | Major |
| Functional | TC34 | Filter Products by Price | Validate that users can filter products based on predefined price ranges, such as low, medium, or high. | The User is logged in and on the products page. | 1. On the product page, open the filter options.<br>2. Select a price range (e.g., $0–$25).<br>3. Apply the filter.<br>4. Verify only products within the selected price range are displayed. | Given the user is logged in on the products page<br>When a price range filter is selected<br>Then only products within that range are displayed | Product list updates to show only items within the selected price range. | Tthe SauceDemo environment does not support price range filters. Retained for portfolio/documentation. | Blocked | Medium | Minor |
| Functional | TC35 | Sort Products Alphabetically | Verify that the user can sort products in alphabetical order from A to Z using the sort dropdown. | The User is logged in and on the products page. | 1. On the product page, open the sort dropdown.<br>2. Select the "Name (A to Z)" option.<br>3. Observe that products are displayed in alphabetical order. | Given the user is logged in and on the products page<br>When the "Name (A to Z)" sort option is selected<br>Then products are displayed in alphabetical order | Products are listed in alphabetical order (A–Z) after applying the sort. | As expected – products were correctly sorted alphabetically. | Passed | Medium | Minor |
| Navigation Post-login | TC36 | Back to Products Button | Validate that the 'Back to Products' button on the product details page correctly redirects the user to the product listing page. | The User is logged in and has navigated to a product details page. | 1. On the product page, click on a product to view its details.<br>2. Click the “Back to Products” button.<br>3. Verify that the user is redirected to the products listing page. | Given the user is on the product details page<br>When the "Back to Products" button is clicked<br>Then the user is redirected to the products listing page | User is successfully redirected from the product details page to the products listing page after clicking the back button. | As expected — redirection works correctly. | Passed | Medium | Minor |
| Usability / Responsiveness | TC37 | Hamburger Menu Functionality | Verify that the hamburger menu opens/closes properly and its options function as expected. | The User is logged in and on a screen where the menu is accessible (e.g., products page). | 1. On the product page, click the hamburger menu icon.<br>2. Confirm the menu opens and displays all expected links.<br>3. Click on each option (e.g., About, Logout) and verify proper redirection or behavior.<br>4. Close the menu and confirm it disappears from view. | Given the user is logged in<br>When the hamburger menu is clicked<br>Then the menu opens and all options function correctly | Menu opens and closes correctly, and each option performs its intended function. | As expected — all actions worked as intended during the test. | Passed | Medium | Major |
| Navigation Post-login | TC38 | Reset App State | Ensure that using the 'Reset App State' option clears the shopping cart and resets product states and UI to default. | The User is logged in and has one or more products added to the cart. | 1. On the product page, add products to the cart.<br>2. Open the hamburger menu.<br>3. Click "Reset App State".<br>4. Open the cart and confirm it is empty.<br>5. Go back to the products page and ensure buttons show "Add to cart". | Given the user is logged in with items in the cart<br>When "Reset App State" is selected from the hamburger menu<br>Then the cart is emptied<br>And all product buttons reset to "Add to cart" | The cart should be cleared and all UI elements (e.g., product buttons) reset to default states. | After clicking “Reset App State”, the cart was emptied and all buttons were reset to "Add to cart", indicating the state was successfully restored. | Passed | Medium | Major |
| Navigation Post-login | TC39 | About Link Navigation | Verify that the 'About' option in the hamburger menu correctly redirects to the official Sauce Labs website. | The User is logged in and on any page where the hamburger menu is available. | 1. On the product page, open the hamburger menu.<br>2. Click the "About" link.<br>3. Verify the browser redirects to https://saucelabs.com/ in a new tab or window. | Given the user is logged in<br>When the "About" link in the hamburger menu is clicked<br>Then the browser redirects to the Sauce Labs homepage in a new tab or window | The "About" link opens the Sauce Labs website in a new tab or window. | Redirection occurred correctly to https://saucelabs.com/. | Passed | Medium | Minor |
| Usability / Responsiveness | TC40 | Responsive Layout Across Pages | Ensure that the application’s layout is fully responsive and maintains usability across desktop, tablet, and mobile devices on critical pages such as products, login, and cart. | The user is logged in. | 1. On the product page, navigate to key pages: products, login, and cart.<br>2. Resize the browser or use device emulation for desktop, tablet, and mobile views.<br>3. Verify the layout adapts responsively and usability is preserved. | Given the user is logged in<br>When the screen is resized to desktop, tablet, or mobile sizes<br>Then the layout adapts responsively<br>And usability is maintained across key pages | Layout adapts smoothly, and usability is preserved across all tested screen sizes. | Responsive behavior confirmed on all tested resolutions. | Passed | Medium | Major |

📜 **Products Page:**
Test product page elements, including browsing, sorting, and filtering. Validate UI consistency and correct item display for all user profiles.

**Prerequisites:**
- User must be logged in as a valid user (e.g., standard_user)
- Access to the Products page: https://www.saucedemo.com/inventory.html
- At least one product must be available on the page for interaction

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Usability / Feedback | TC41 | Feedback on Duplicate Add to Cart | Ensure the system provides clear feedback or disables the 'Add to Cart' button when the user attempts to add a product already in the cart. | The user is logged in. | 1. On the product page, add a product to the cart.<br>2. Attempt to add the same product again.<br>3. Observe the button state or feedback message. | Given the item is already in the cart<br>When the user attempts to add it again<br>Then the 'Add to Cart' button is disabled or a message indicates the item is already added | Button disables or feedback message shows preventing duplicate additions. | Button state changed or feedback displayed as expected. | Passed | Medium | Major |
| Security / Session | TC42 | Session Expiration Behavior | Ensure the system securely handles session expiration by redirecting unauthenticated users to the login page when accessing protected resources. | The user is logged in. | 1. Log in as a standard user.<br>2. Close the browser tab to simulate session expiration.<br>3. Try to access a protected page without logging in again.<br>4. Confirm redirection to the login page. | Given the user's session has expired<br>When a protected page is accessed<br>Then the user is redirected to the login page | User is redirected to login upon session expiration. | Upon session expiration, user is correctly redirected to login, maintaining security integrity. | Passed | Medium | Critical |
| Navigation | TC43 | Verify Logout Functionality | Verify that clicking the logout button properly ends the user session and redirects to the login page. | The user is logged in. | 1. Log in as a standard user.<br>2. Click the burger menu icon.<br>3. Click the "Logout" option.<br>4. Confirm redirection to the login page. | Given the user is logged in<br>When the logout button is clicked<br>Then the user is redirected to the login page | The user session ends and the system redirects to the login page, preventing access to authenticated areas. | Logout and redirection worked as expected. | Passed | Medium | Critical |
| Usability / Display | TC44 | Verify Product Price Display Format | Ensure product prices are displayed consistently and clearly across the product listing page. | The User is logged in and on the products page. | 1. On the product page, navigate to the products page.<br>2. Check that prices for all products are clearly visible and formatted consistently. | Given the user is logged in and on the products page<br>When product prices are viewed<br>Then all prices are displayed clearly and consistently with correct formatting | Product prices are visible with consistent currency format (e.g., "$29.99"), clear font, and proper alignment. | All product prices were displayed consistently and clearly. | Passed | Medium | Minor |

🛒 **Shopping Cart:**
Verify cart behavior for adding/removing items, updating quantities, and clearing cart after logout. Confirm correct UI and button actions.

**Prerequisites:**
- User must be logged in
- User must have access to the Cart page: https://www.saucedemo.com/cart.html
- At least one product must be added to the cart for some tests

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Functional | TC50 | Add single item to cart | Verify that a product can be successfully added to the shopping cart, and the cart state updates accordingly. | The User is logged in and on the products page. | 1. Navigate to the products page.<br>2. Click 'Add to cart' on a product.<br>3. Verify button changes state (e.g., to "Remove").<br>4. Click cart icon.<br>5. Verify item appears in cart and cart count updates. | Given the user is logged in on the products page<br>When 'Add to cart' is clicked for a product<br>Then the button changes to 'Remove'<br>And the item is added to the cart<br>And the cart icon count increases by one | Item is successfully added to the cart; UI elements update to reflect the addition, including button state and cart count. | The item is successfully added to the cart, and the UI elements (button state and cart count) are updated to reflect the addition. | Passed | Medium | Critical |
| Functional | TC51 | Remove item and update cart count and total price | Verify that removing an item from the shopping cart updates the cart contents, item count, and total price correctly. | The User is logged in and has multiple items in the shopping cart. | 1. Navigate to cart.<br>2. Add products to cart.<br>3. Click 'Remove' on an item.<br>3. Verify cart icon count and total price update. | Given the user has multiple items in the cart<br>When an item is removed<br>Then the item is removed from the cart<br>And the cart icon count decreases by one<br>And the total price updates accordingly | Item removed; cart count and total price reflect the change. | The item is removed, and the cart count and total price are updated correctly. | Passed | Medium | Major |
| Functional | TC52 | Add multiple distinct items to cart | Verify that multiple distinct products can be added to the shopping cart and all appear correctly when viewing the cart. | The User is logged in and on the products page. | 1. Add two or more distinct products to the cart.<br>2. Open the cart page.<br>3. Verify that all added items are present. | Given the user has added multiple distinct items to the cart<br>When the cart is viewed<br>Then all added items are listed correctly | All distinct items appear in the cart without omission. | All distinct items appear in the cart without omission. | Passed | Medium | Major |
| Functional | TC53 | Check cart icon count | Verify that the cart icon correctly displays the current number of items added to the cart. | The User is logged in with no items added to the cart. | 1. Navigate to the products page.<br>2. Add products to the cart.<br>3. Observe the cart icon count.<br>4. Verify that the number shown matches the number of items added. | Given the user has items in the cart<br>When the cart icon is viewed<br>Then the icon displays the correct item count | The cart icon accurately shows the total number of items in the cart. | The cart icon accurately shows the total number of items in the cart. | Passed | Medium | Major |
| Functional / Usability / UX | TC54 | Access empty cart | Verify the cart page shows a proper empty state when no items are added. | The User is logged in with no items added to the cart. | 1. Log in with an empty cart.<br>2. Click the cart icon to open the cart page.<br>3. Verify no products are displayed.<br>4. Confirm an empty cart message or layout is visible.<br>5. Check that the "Checkout" button is hidden or disabled. | Given the cart is empty<br>When the cart page is opened<br>Then an empty cart message is displayed<br>And the "Checkout" button is hidden or disabled | Cart page shows no products, displays a clear empty cart message, and the "Checkout" button is hidden or disabled. | The cart shows no products, but the "Checkout" button stays visible and active with no empty cart message, which may confuse users.<br>Suggestion: Hide or disable the "Checkout" button when the cart is empty and add a clear empty cart message to improve user | Passed | Medium | Major |
| Functional | TC55 | Persist Cart After Logout | Verify that items added to the cart remain present after the user logs out and logs back in, ensuring session persistence. | The User is logged in and has added items to the cart. | 1. Add one or more items to the cart.<br>2. Log out.<br>3. Log back in with the same user credentials.<br>4. Open the cart page and confirm that the items remain. | Given the user has items in the cart<br>When they log out and log back in<br>Then the cart retains the previously added items | The cart retains items after logout and login, confirming session persistence. | The cart retains previously added items after logout/login, demonstrating reliable session persistence. | Passed | Medium | Major |
| Functional | TC56 | Checkout button visible | Verify that the checkout button is displayed and accessible when the cart contains one or more items. | The User is logged in and has at least one item in the shopping cart. | 1. Add products to the cart.<br>2. Navigate to the cart page.<br>3. Verify the checkout button is visible and enabled. | Given the user has items in the cart<br>When the cart page is opened<br>Then the checkout button is visible and enabled | Checkout button is visible and enabled, allowing the user to proceed with the purchase. | Checkout button is visible and enabled when the cart contains items. | Passed | Medium | Major |
| Functional | TC57 | Add item disables "Add" button | Verify that after adding a product, the 'Add to cart' button changes state to prevent duplicate additions. | The User is logged in and on the products page. | 1. Navigate to the products page.<br>2. Click 'Add to cart' on a product.<br>3. Confirm the button state changes to 'Remove' or becomes disabled. | Given the user is on the products page<br>When they click 'Add to cart' on a product<br>Then the button updates to 'Remove' or is disabled | The 'Add to cart' button visually updates to indicate the item is added and prevents duplicate adds. | The button state changes as expected, preventing further additions without removal. | Passed | Medium | Major |
| Functional | TC58 | Verify cart total updates dynamically | Validate that the cart's total price updates dynamically and correctly after adding or removing products. | The User is logged in and on the products page. | 1. Add one or more products to the cart.<br>2. Observe the total price displayed in the cart or cart icon.<br>3. Remove one or more items from the cart.<br>4. Observe the total price updates immediately to reflect the change. | Given the user has items in the cart<br>When the user add or remove an item<br>Then the cart total price updates immediately to reflect the change | Cart total price updates immediately after each add/remove. | Test blocked because the application UI does not show cart total price, only item count, making it impossible to verify dynamic total updates. | Blocked | Medium | Major |
| User Session | TC59 | Cart persistence across multiple tabs | Validate that the cart state is synchronized in real time across multiple open browser tabs for the same user session. | The User is logged in and on the products page. | 1. Open two browser tabs on the products page.<br>2. Add an item to the cart in Tab A.<br>3. Verify Tab B updates cart contents automatically.<br>5. Remove item from cart in Tab B.<br>6. Verify Tab A updates automatically. | Given the user has multiple tabs open on the products page<br>When the cart is modified in one tab<br>Then all other tabs update the cart state immediately | Cart content synchronizes instantly across all open tabs without manual refresh. | The cart does not synchronize in real time across multiple tabs.<br>Users must manually refresh to see updated cart contents, which degrades user experience. | Failed | Medium | Major |
| Functional Limits | TC60 | Max items limit in cart | Verify system behavior when the user attempts to exceed the cart's item limit, either through quantity or number of unique products. | The User is logged in and on the products page. | 1. On the products page, add all available unique items to the cart.<br>2. Attempt to add duplicates of the same items multiple times.<br>3. Observe whether the system limits the number of items per cart or per product.<br>4. Check for error messages, blocking behavior, or item cap. | Given the user is logged in on the products page<br>When multiple items are added continuously to the cart<br>Then the system enforces a maximum cart size or shows a warning if the limit is reached | The system should prevent the user from exceeding the maximum allowed items in the cart, either by blocking additions or displaying an error message. | SauceDemo does not support quantity control or a large enough catalog to simulate cart limits. No max limit enforcement could be observed. | Blocked | Medium | Minor |

**🛒 Shopping Cart:**
Verify cart behavior for adding/removing items, updating quantities, and clearing cart after logout. Confirm correct UI and button actions.

**Prerequisites:**
- User must be logged in
- User must have access to the Cart page: https://www.saucedemo.com/cart.html
- At least one product must be added to the cart for some tests

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| User Session | TC61 | Cart persistence after long browser close | Verify whether the cart contents are preserved after the browser is completely closed and reopened. | The User is logged in and have at least one item added to the cart before closing the browser. | 1. Add one or more items to the cart.<br>2. Close the browser completely (not just the tab).<br>3. Wait a defined period to simulate a “long time.”<br>4. Reopen the browser and log in with the same user.<br>5. Verify if the previously added items remain in the cart. | Given the user added items to the cart and closed the browser<br>When the browser is reopened and the user logs in after a delay<br>Then the cart either retains or clears the items based on system behavior | The system should persist cart items across sessions after the user logs in again. | After reopening the browser and logging in, the previously added items remained in the cart, confirming persistence of cart data across sessions. | Passed | Medium | Major |
| Functional | TC62 | Clear cart functionality | Verify that the user can remove all items from the shopping cart at once, and that the cart updates accordingly. | The User is logged in and have one or more items already added to the cart. | 1. Go to the cart with items already added.<br>2. Remove all items from the cart one by one using the available "Remove" buttons.<br>3. Verify that the cart becomes empty and the UI reflects the empty state. | Given the user has items in the cart<br>When the user removes all items<br>Then the cart is empty<br>And the UI displays an empty state (e.g., message or zero item count) | All items are removed from the cart and the interface shows an empty cart state (e.g., “Your cart is empty”, item count = 0). | All items were successfully removed from the cart. The UI updated correctly, showing the empty state. | Passed | Medium | Major |
| Usability / Responsiveness | TC63 | Cart page responsiveness | Verify that the cart page layout adapts correctly across different screen sizes (mobile, tablet, desktop), maintaining usability and visual integrity. | The User is logged in and able to access the cart page on devices with varying screen sizes. | 1. Log in and navigate to the cart page.<br>2. Simulate mobile view (e.g., using Chrome DevTools or an actual device).<br>3. Observe if the layout adapts (responsive breakpoints).<br>4. Verify button visibility, alignment, and readability.<br>5. Switch to desktop view and confirm consistent design. | Given the user is logged in and on the cart page<br>When the screen size changes (mobile, tablet, desktop)<br>Then the cart layout adapts responsively<br>And all elements remain visible and functional | Cart layout should respond to screen size changes, preserving readability, button accessibility, and overall usability. | The cart page adapted properly across tested screen sizes. Layout, buttons, and content remained functional and well-aligned. | Passed | Medium | Major |
| Negative Test | TC64 | Clear Cart After Logout | Verify that the shopping cart is cleared when the user logs out and logs back in, ensuring no data persistence across sessions. | The User is logged in and has added items to the cart prior to logout. | 1. Add one or more items to the cart.<br>2. Log out from the application.<br>3. Log back in with the same user credentials.<br>4. Navigate to the cart page.<br>5. Verify the cart is empty. | Given the user is logged in with items in the cart<br>When the user logs out and logs back in<br>Then the cart is emptied to prevent session persistence | Cart should be cleared after logout, so the user starts with an empty cart upon new login. | The cart retained the items after logout and subsequent login, showing persistence that may not be expected. | Failed | High | Major |
| Security / Access Control | TC65 | Access cart without login | Verify that unauthorized users are prevented from accessing the cart page and are redirected to the login page with an appropriate message. | The User is not logged in. | 1. Open browser without being logged in.<br>2. Navigate directly to the cart page URL: https://www.saucedemo.com/cart.html.<br>3. Observe whether the user is redirected to the login page.<br>4. Check for the presence of an appropriate warning or info message. | Given the user is not logged in<br>When the cart page URL is accessed directly<br>Then the user is redirected to the login page<br>And a message indicates login is required | Unauthorized users are redirected to the login page with a clear message explaining that login is required to access the cart. | User was redirected to login page with message:<br>"You can only access '/cart.html' when you are logged in." | Passed | High | Major |
| Functional | TC66 | Refresh after removing item | Verify that items removed from the cart remain removed after the page is refreshed, ensuring cart state persistence. | The User is logged in and at least one item has been removed from the cart. | 1. Navigate to the cart page.<br>2. Remove an item from the cart.<br>3. Refresh the cart page.<br>4. Verify that the removed item does not reappear. | Given the user has removed an item from the cart<br>When the cart page is refreshed<br>Then the removed item is absent<br>And the cart shows the updated items | The cart should maintain the updated state with removed items excluded after a page refresh. | Removed items stayed removed after refresh. | Passed | Medium | Major |
| Negative Test / Network Issues | TC67 | Cart Interaction During Offline State | Verify that the system properly detects offline status and handles add/remove cart actions by showing errors or queuing actions, and disables checkout while offline. | The User has internet connection initially, is logged in, and has items in cart or is browsing products. User then disconnects internet. | 1. Log in and add items to cart or browse products.<br>2. Disconnect the device from the internet.<br>3. Attempt to add a new item to the cart.<br>4. Attempt to remove an existing item from the cart.<br>5. Observe system responses and UI messages.<br>6. Check if checkout button is disabled or blocked. | Given the user is offline<br>When attempting to add or remove cart items<br>Then an error message is shown or the action is queued<br>And the checkout button is disabled | The system prevents cart modifications while offline, provides clear feedback or queues actions, and disables checkout to avoid data inconsistency. | The system currently allows add/remove actions without any error or offline warning, which may cause data loss or inconsistent UX. Implement offline detection to block or queue cart actions and disable checkout, ensuring data consistency and clear user communication. | Failed | High | Critical |
| Negative Test | TC68 | Add invalid item quantity | Verify that the system validates item quantity input and prevents zero or negative values from being added. | The User is logged in. | 1. Navigate to the product or cart page.<br>2. Attempt to add an item with zero quantity.<br>3. Attempt to add an item with negative quantity.<br>4. Observe error messages or system behavior. | Given the user tries to add an item with a quantity of zero or a negative quantity<br>When the "Add to Cart" action is performed<br>Then the system should display an error message<br>And the item should not be added to the cart | The system must display an error message and prevent adding invalid item quantities. | Not executable due to the system not allowing quantity edits; test blocked by environment limitations. | Blocked | Medium | Major |

💳 **Checkout Process:**
Ensure proper flow through checkout steps: information entry, overview validation, and final order completion. Validate field requirements and error handling.

**Prerequisites:**
- User must be logged in
- User must have at least one product in the cart
- Access to checkout pages:
  - Step One: https://www.saucedemo.com/checkout-step-one.html
  - Step Two: https://www.saucedemo.com/checkout-step-two.html
  - Complete: https://www.saucedemo.com/checkout-complete.html
- Test user must have valid information to fill in checkout fields

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Checkout Flow | TC71 | Access Checkout Page | Verify that clicking the “Checkout” button on the cart page redirects the user to the checkout information form (Step One), initiating the checkout process. | The User is logged in and has at least one item in the cart. | 1. On the products page, add one or more items to the cart.<br>2. Click the cart icon to open the cart.<br>3. Click the “Checkout” button.<br>4. Confirm redirection to `/checkout-step-one.html`.<br>5. Check that First Name, Last Name, and Postal Code fields are displayed. | Given the user is logged in and has items in the cart<br>When the “Checkout” button is clicked on the cart page<br>Then the system redirects to the first step of checkout<br>And displays the required personal information fields | User is redirected to `/checkout-step-one.html`, where First Name, Last Name, and Postal Code fields are presented for input. | As expected. Redirection successful and personal information fields are displayed. | Passed | Medium | Critical |
| Field Validation / Negative Test | TC72 | Required Fields Validation at Checkout | Verify that all mandatory fields (First Name, Last Name, Postal Code) must be completed to proceed with checkout. | The User is on the first step of the checkout process (/checkout-step-one.html). | 1. Leave First Name, Last Name, and Postal Code fields empty.<br>2. Click the “Continue” button.<br>3. Observe error messages displayed for required fields. | Given the user is on the checkout information page<br>When they attempt to continue without filling required fields<br>Then error messages appear for all mandatory fields<br>And progression to the next step is blocked | The system prevents progression and displays error messages for First Name, Last Name, and Postal Code. | Only “First Name” shows an error. Other required fields do not. Suggestion: Implement validation and user feedback for all required fields to improve clarity and prevent incomplete submissions. | Passed | High | Critical |
| Functional Flow / Checkout Completion | TC73 | Complete Checkout Flow with Valid Data | Ensure that a user can successfully complete the checkout process from information entry to final confirmation using valid data. | The User is logged in, has one or more items in the cart, and is on the first step of the checkout process. | 1. Enter valid values in the First Name, Last Name, and Postal Code fields.<br>2. Click “Continue”.<br>3. Review item details and totals.<br>4. Click “Finish” to finalize the order.<br>5. Verify that the confirmation page is displayed with a success message. | Given the user is logged in with items in the cart<br>And has entered valid personal information<br>When they complete the checkout flow by clicking “Continue” and then “Finish”<br>Then the order is completed<br>And a confirmation message with the final page is displayed | User is redirected to the order confirmation page, with a success message indicating completion. | Order completed successfully. Confirmation message and final screen were shown. | Passed | High | Critical |
| Navigation / Usability | TC74 | Cancel Checkout and Return to Cart | Ensure that the user can cancel the checkout process and return to the cart without losing any previously added items. | The User is logged in, has items in the cart, and is on the first step of the checkout process. | 1. On the checkout step one page, click the “Cancel” button.<br>2. Verify redirection to the cart page.<br>3. Confirm that all items added before entering checkout are still present. | Given the user is on the first checkout step<br>And has items in the cart<br>When they click the “Cancel” button<br>Then they are redirected to the cart page<br>And the cart retains all items | User is redirected to the cart page, and all cart items remain intact. | As expected. The user returned to the cart and all items were preserved. | Passed | Medium | Critical |
| Field Validation / Negative Test | TC75 | Checkout Form Input Validation | Ensure that invalid entries in checkout fields are detected, blocked, and accompanied by meaningful error messages. | The User is logged in, has items in the cart, and is on the checkout step one page. | 1. Enter digits or symbols in First Name and Last Name fields.<br>2. Enter an invalid Postal Code (e.g., "abc", "!@#").<br>3. Click the “Continue” button.<br>4. Observe whether the form is submitted or blocked. | Given the user is on the checkout information page<br>When they enter invalid data in name or postal code fields<br>And click the “Continue” button<br>Then the system should display error messages<br>And prevent moving forward in the checkout process | Submission is blocked. Each invalid field displays a contextual error message. | The system allows the user to proceed to the next checkout step despite invalid inputs. No validation errors are displayed. | Failed | High | Critical |
| Navigation / Usability | TC76 | Navigation Between Checkout Steps | Ensure that navigating back and forth during the checkout flow does not erase form data or reset the cart contents. | The User is logged in, has items in the cart, and has reached the second checkout step. | 1. On the checkout step one page, fill in valid First Name, Last Name, and Postal Code.<br>2. Click “Continue” to go to the order summary page.<br>3. Click the “Back” button.<br>4. Verify that the form still displays the previously entered data.<br>5. Confirm that the cart still contains the same products. | Given the user is on the order summary page<br>When they click “Back”<br>Then they should return to the form page<br>And all previously entered values should remain visible<br>And the cart should still contain the added items | User data is retained in the form, and cart contents are preserved. | As expected. Navigating back does not reset form or cart. | Passed | Medium | Major |
| Negative Test / Business Rule | TC77 | Empty Cart Checkout Prevention | Ensure the checkout process cannot be started if the cart has no items. | The User is logged in and the cart is empty. | 1. Ensure there are no products in the cart.<br>2. Attempt to access the checkout page:<br>- By clicking the “Checkout” button (if present).<br>- Or by navigating to `/checkout-step-one.html` directly.<br>3. Observe system behavior. | Given the user has no items in the cart<br>When they try to access the checkout page<br>Then the system should prevent access<br>And inform the user that items are required to proceed | System blocks access to checkout when cart is empty. An appropriate message is shown or the user is redirected to the cart/products page. | User is allowed to access the checkout page despite the cart being empty. No warning or redirection occurs. | Failed | High | Critical |
| Confirmation / UX | TC78 | Confirmation Message after Checkout | Ensure that after a successful checkout, the user is presented with a clear and friendly confirmation message. | The checkout process has been completed. | 1. Complete checkout by entering valid data and clicking “Finish”.<br>2. Wait for the confirmation page to load.<br>3. Observe the confirmation message displayed to the user. | Given the user has completed checkout successfully<br>When the confirmation page is loaded<br>Then the system should display a friendly success message confirming the order | A clear, visible message confirms the order success (e.g., “Thank you for your order!”). | As expected. Confirmation message is shown clearly. | Passed | Medium | Major |
| Usability / Post-checkout Navigation | TC79 | Confirmation Page Buttons Functionality | Ensure buttons on the confirmation page, specifically “Back Home,” function correctly and redirect the user to the products page. | The User is on the order confirmation page after completing checkout. | 1. Click the “Back Home” button on the confirmation page.<br>2. Verify the user is redirected to the products listing page. | Given the user is on the confirmation page<br>When they click “Back Home”<br>Then they are redirected to the products page | User is successfully redirected back to the product catalog. | As expected, redirection occurs correctly. | Passed | Medium | Major |
| UI Behavior / Usability | TC80 | Checkout Button Availability Based on Cart Content | Ensure that the “Checkout” button is enabled only when the cart contains one or more items. | The User is logged in and on the cart page. | 1. Verify cart is empty.<br>2. Check the state of the “Checkout” button (should be disabled or hidden).<br>3. Add an item to the cart.<br>4. Return to the cart page and verify the “Checkout” button is enabled. | Given the user is on the cart page<br>When the cart is empty<br>Then the “Checkout” button should be disabled or hidden<br>And when there is at least one item<br>Then the button should be enabled and clickable | The “Checkout” button is disabled or hidden if the cart is empty, and enabled when there are items. | The “Checkout” button remains active and clickable regardless of cart contents. | Failed | High | Major |

🔁 **Core Functionality:**
Validate overall platform stability: navigation, logout behavior, session handling, and basic accessibility like keyboard navigation.

**Prerequisites:**
- User must be logged in
- Access to key pages:
- Products: https://www.saucedemo.com/inventory.html
- Cart: https://www.saucedemo.com/cart.html
- Browser should support keyboard navigation (for accessibility tests)
- Session should be clean (fresh login) for session handling validations

| Category | ID | Test Name | Objective | Precondition | Steps | BDD (Gherkin Syntax) | Expected Result | Actual Result (Occurrence) | Status | Priority | Severity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Performance | TC81 | Page Load Time | Measure the homepage's load time to ensure it meets performance benchmarks under normal conditions. | The User is not logged in and browser cache is cleared (optional). | 1. Open the browser in incognito/private mode.<br>2. Navigate to https://www.saucedemo.com/<br>3. Start timing when the request begins.<br>4. Record the time when the page finishes loading. | Given the user is on a fresh browser session<br>When the SauceDemo homepage loads<br>Then the page load time does not exceed 3 seconds under normal network conditions | Page load completes within acceptable performance threshold (under 3 seconds on standard network). | The homepage loads fully within 3 seconds under normal network conditions, meeting the defined performance benchmark without errors or noticeable delays. | Passed | Medium | Minor |
| Usability / Accessibility | TC82 | Keyboard Navigation Accessibility | Verify that all interactive elements (inputs, buttons, links) are accessible via keyboard navigation alone, using Tab and Enter keys. | The User is logged in to access the product page, or on the login page. | 1. Ensure browser focus is on the page (not dev tools).<br>2. Press Tab repeatedly to navigate through all focusable elements.<br>3. Confirm logical order of focus (inputs, buttons, links).<br>4. Use Enter key to trigger any active element (e.g., Login button, Add to Cart). | Given the user is on the login or products page<br>When the user navigates using the Tab key and activates elements with Enter<br>Then focus moves in logical order<br>And all elements respond correctly to keyboard input | Keyboard navigation works logically; elements activated via Enter key. | All elements are reachable and respond via keyboard. However, the visual focus indicator (outline) is weak and may hinder usability for keyboard users. CSS improvement recommended. | Passed | Medium | Major |
| Accessibility | TC83 | Color Contrast | Ensure that text elements on the product page maintain proper contrast for readability. | The User is logged in and on the products page. | 1. On the products page, identify visible text (product names, prices, descriptions, menu items).<br>2. Use a contrast analysis tool (e.g., Chrome DevTools → Accessibility tab, or WebAIM Contrast Checker).<br>3. Check if text color vs background color meets required contrast ratios.<br>4. Record any failures. | Given the user is on the products page<br>When they evaluate the contrast between text and background<br>Then all text should be clearly readable | Contrast was adequate across key elements. | Tested using Chrome DevTools contrast tool. No failures detected. | Passed | Medium | Major |
| Usability / Responsiveness | TC84 | Responsive Layout Across Pages | Ensure that all core pages (login, products, cart) render correctly and remain usable across desktop, tablet, and mobile screen resolutions. | The User must be able to access login, product, and cart pages. | 1. Open the application in a browser.<br>2. Go to login, products, and cart pages.<br>3. Resize the window to desktop, tablet, and mobile sizes.<br>4. Check if layout adjusts and remains usable. | Given the user resizes the screen<br>When accessing login, products, or cart pages<br>Then the layout should adjust correctly to the screen size | Layout adapts smoothly across all tested breakpoints. No visual or functional issues observed. | The layout adapts seamlessly across desktop, tablet, and mobile resolutions on login, products, and cart pages, with no visual distortions or usability issues detected. | Passed | High | Major |
| Session & Cookies | TC85 | Cart Persistence After Refresh | Verify that cart contents persist after refreshing the products or cart page. | The User is logged in and has at least one product added to the cart. | 1. On the products page, add one or more products to the cart.<br>2. Refresh the page (e.g., /inventory or /cart).<br>3. Open the cart and verify that the items are still there. | Given the user has added items to the cart<br>When the page is refreshed<br>Then the cart should retain its contents | Cart items persisted correctly after page refresh. No loss of data observed. | Cart contents remain intact and correctly displayed after refreshing the products or cart page, with no loss of items or data. | Passed | Medium | Critical |
| Error Handling & Recovery | TC86 | Behavior with Invalid URLs | Verify that accessing an invalid URL results in a redirect to a valid page or a user-friendly error message. | The User is not logged in. | 1. Open the browser and navigate to https://www.saucedemo.com/.<br>2. In the address bar, enter a non-existent path (e.g., /invalidpage).<br>3. Press Enter and observe the page behavior. | Given the user is not logged in<br>When they navigate to an invalid URL<br>Then the system should show a clear error message or redirect to the login page | Redirect to login page or display a user-friendly error message. | The page remained blank with no feedback or redirection. | Failed | Medium | Major |
| Cross-Platform Compatibility | TC87 | Cross-Browser Compatibility | Ensure that login and basic actions (add to cart) behave consistently on Chrome, Firefox, and Edge. | The User has access to browsers. | 1. Open SauceDemo in Chrome, Firefox, and Edge.<br>2. Log in with valid user credentials.<br>3. Add a product to the cart.<br>4. Verify the actions behave consistently across browsers. | Given the user opens the app in a browser<br>When they perform basic actions like login and adding products<br>Then the behavior should be consistent across all tested browsers | Behavior is consistent across Chrome, Firefox, and Edge. | Login and add-to-cart actions function consistently and without issues across Chrome, Firefox, and Edge browsers. | Passed | Medium | Minor |