## Technical task

**Estimation**: *one week* [03.05.2023, 11.30 PM – 10.05.2023, 11.30 PM]

**Objectives:** functional testing of a test site (softwaretestingboard.com)

**Resources required:** SeleniumWebdriverIO & Cucumber

### **Testing Strategy**

**Defined User Stories and planned possible scenarios:**

**User story (US)1: Navigation**

**Priority: High, Severity: Medium**

As a user, I want to be able to navigate the Luma website easily and find the products I'm looking for quickly, so that I can make a purchase without any hassle.

**US2: Search functionality**

**Priority: High, Severity: Medium**

As a user, I want to be able to search for products on the Luma website and see accurate results, so that I can find the products I want without having to browse through multiple pages.

**US3: Product listing and details**

**Priority: High, Severity: Medium**

As a user, I want to be able to view accurate product details such as prices, descriptions, and images, so that I can make an informed decision when purchasing products.

**US4: Add to cart and checkout**

**Priority: High, Severity: High**

As a user, I want to be able to add products to my cart and checkout quickly and easily, so that I can complete my purchase without any delays.

**US5: Account creation and login**

**Priority: High, Severity: Medium**

As a user, I want to be able to create an account and log in to the Luma website easily, so that I can view my order history and make future purchases more conveniently.

**US6: Payment processing**

**Priority: High, Severity: High**

As a user, I want to be able to make payments for my purchases easily and securely, so that I can complete my purchase without any security concerns.

**US7: Responsive design**

**Priority: Medium, Severity: Low**

As a user, I want to be able to view the Luma website on different devices and screen sizes without any issues, so that I can make a purchase regardless of the device I'm using.

**US8: Error handling**

**Priority: Medium, Severity: High**

As a user, I want to be informed of any errors or issues that occur while using the Luma website, so that I can take appropriate action to resolve the issue and complete my purchase without any further delays."

### Test Plan

**Requirement** for the **testing scope**: 3-5 end to end tests as minimum.

Based on the ***priority*** and ***severity*** ratings, the top 5 US to prioritized for testing are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Story | Name | Priority | Severity | Dependency |
| US4 | Add to cart and checkout | High | High | US5, (US6\* – not in our case) |
| US6\* | Payment processing | High | High | US5 |
| US1 | Navigation | High | Medium |  |
| US2 | Search functionality | High | Medium |  |
| US5 | Account creation and login | High | Medium |  |
| US3 | Product listing and details | High | Medium |  |

\*see the comments below for that user story

**US4:Add to cart and checkout**

***Test scenarios:***

US4 – Test scenario (TS)1 Test adding products to the cart and verify that the correct products and quantities are added (scenario – happy path).

US4-TS2 – Test subtract all products from the cart and verify that couldn’t checkout with empty basket (negative path).

**US6:Payment processing – NO TESTS PREPARED, because the example site has no valid payment module for entering data**

***Test scenarios:***

US6-TS1 – Verify that users can complete their purchases after entering payment data without any issues (happy path).

US6-TS2 – Verify that users can’t complete their purchases without payment data (negative path).

**US1: Navigation**

***Test scenarios:***

US1-TS1 – Navigate to sub-pages for each menu item (happy path).

US1-TS2 – Verify users cannot place a product in the basket without entering size and color (negative path).

**US2:Search functionality**

***Test scenarios:***

US2-TS1 – Test the search functionality to ensure accurate search results (happy path).

US2-TS2 – Prevent navigation to non-existing product (negative path)

**US5:Account creation and login**

***Test scenarios:***

US5-TS1 – Test the account creation process and verify that users can create accounts (happy path).

US5-TS2 – Test that users can not create accounts when all necessary inputs were not done (negative path).

**US3:Product listing and details**

***Test scenarios:***

US3-TS1 – Test that users can view more details for available products (happy path)

US3-TS2 – Test that users cannot view related products in Sale menu if in the product category there are not products with discount price (negative path).

### **Bug report related to US3-TSC2**

**Title**: Products which are not in sale from separate category are showing in main Sale page

**Description**: On the main Sale menu page, related products are showing for a specific category (Women/Men – > sub category Pants) even though there are not products with discount prices in that category. However, when checking the main Sale page, these products are listed as discounted and appearing incorrectly.

**Steps to Reproduce:**

1) Go to the main Sale menu page;

2) Navigate to a product category Women/Men and look for category Pants. Observe that product type is shown women’s/men’s deals categories of the Sales page menu (Please see Snapshot 1);

3) Go to the individual product category Pants in Women/Men menus;

4) Go to filter Sales and check availability for products in sale. Observe that we do not have product Pants with a discount price in category sale (Please see Snapshots 2 & 3).

**Expected result**: The Sale menu page should show all related products for every specific category that have products with discount prices and main page do not list any sales if there are not the same in the categories pages.

**Actual result**: The Sale menu page is showing related products Pants for a category Women/Men that has no discounted products.

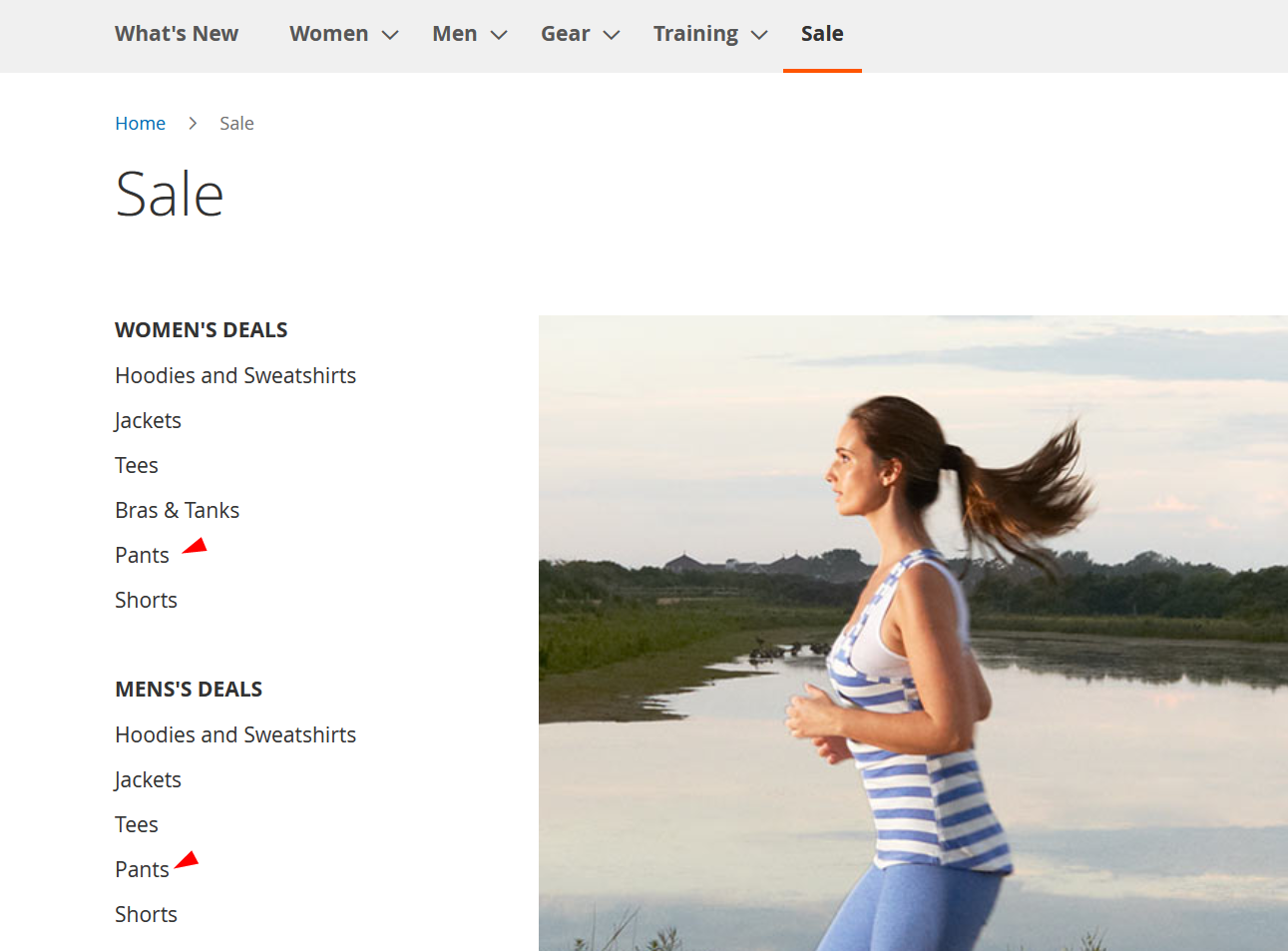
**Severity**: Medium. The issue is limited to specific product categories, but could result in user experience (missing out on discounts) and negatively impact sales.

**Priority**: Project manager is assigned to the bug.

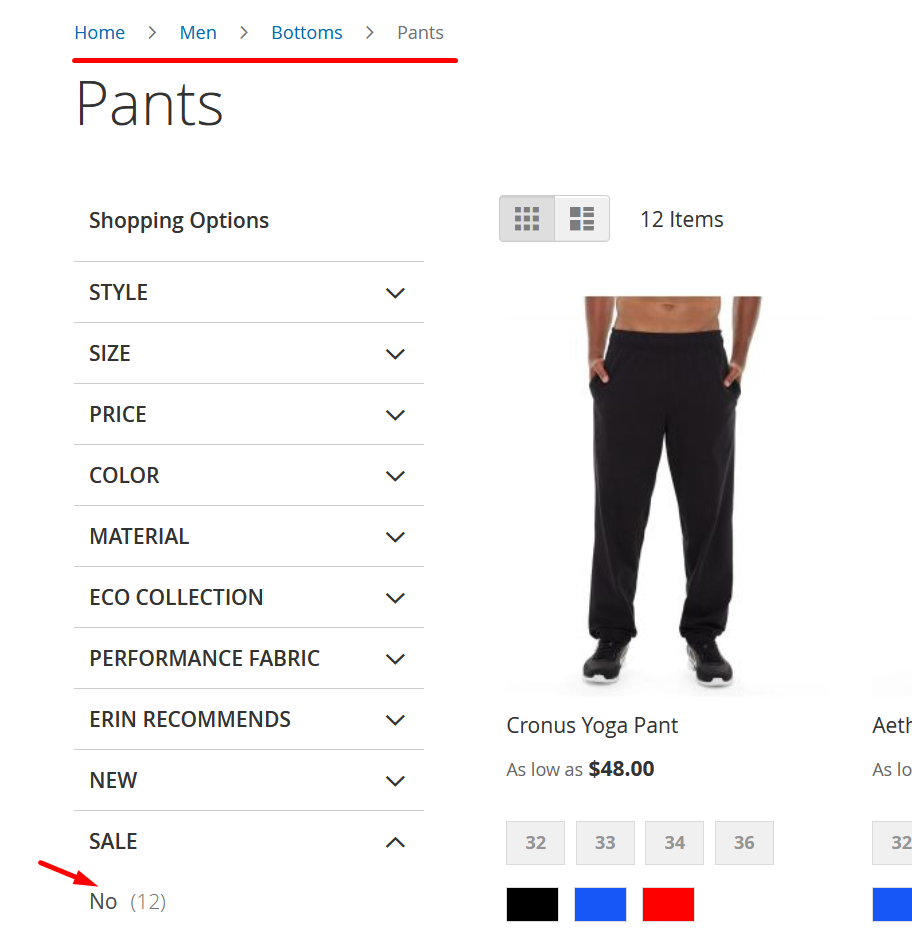
**Environment**: OS Windows 11 Home, Firefox version 112.0.2, site version: test version.

**Attachments**:

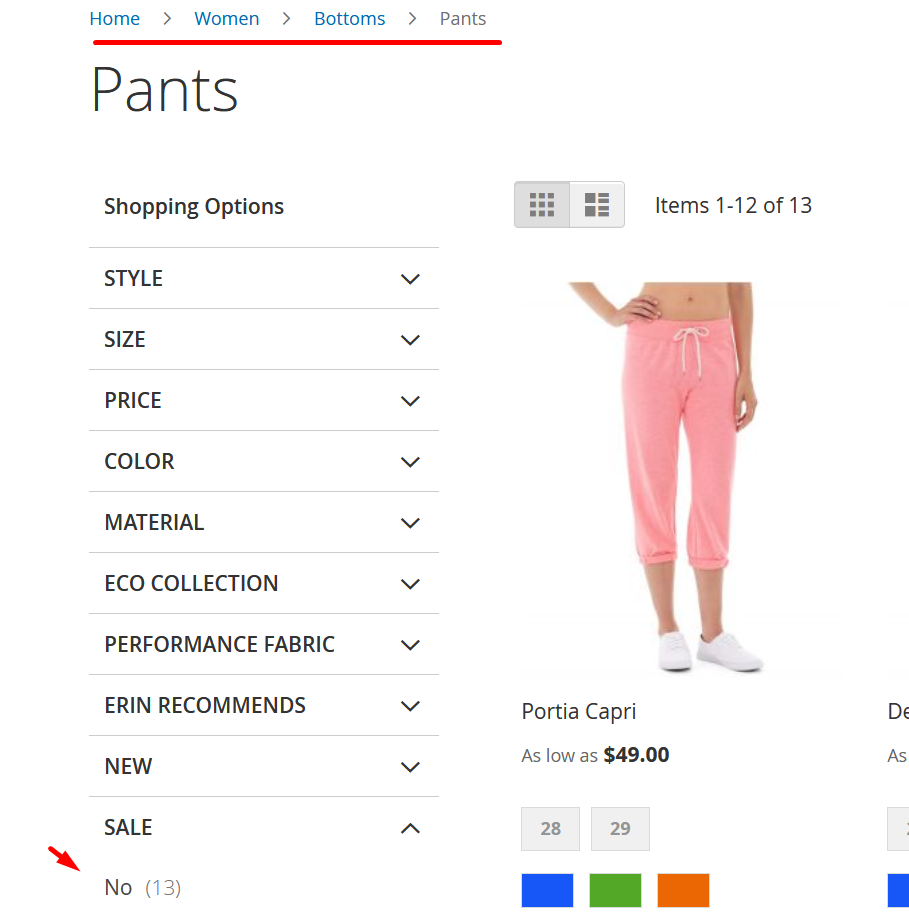
Snapshot 1



Snapshot 2



Snapshot 3



### Test cases

**Framework**: Selenium Webdriver IO (version 4.9.0), Cucumber (version ), Code editor – VSC (version: 1.77.3, Chromium: 102.0.5005.196), Node.js: 16.14.2, V8: 10.2.154.26-electron.0, OS: Windows\_NT x64 10.0.23419

**Environment**: Firefox (version 112.0.2)

**Test object**: softwaretestingboard.com

link to the files: https://github.com/seyrekov/JavaScript/tree/main/Selenium/features

**NOTE**: Code content was provided as a part of the test plan, as follows below. The reason the code be provided here is the additional assurance that technical task was done in time, in case of disturbed access to GitHub by some reason.

### **US1-TS1-Navigation-HP.feature**

1. **Feature: Navigation**

As a user, I want to be able to navigate the Luma website easily and find the products I'm looking for quickly, so that I can make a

purchase without any hassle.

**Scenario**: Navigate to sub-pages for each menu item (happy path)

Given the user is on the home page of the Luma website

When they click on the "What's New" menu item

Then they should see the "What's New" page

When they click on the "Men" menu item

Then they should see the "Men" page

When they click on the "Women" menu item

Then they should see the "Women" page

When they click on the "Gear" menu item

Then they should see the "Gear" page

When they click on the "Training" menu item

Then they should see the "Training" page

When they click on the "Sale" menu item

Then they should see the "Sale" page

### **US1-TS1-Navigation-HP.js**

const { Given, When, Then } = require('cucumber')

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('I am on the home page', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('I click on the {string} menu item', async function (menuItem) {

  switch (menuItem) {

    case "What's New":

      await driver.findElement(By.css('#ui-id-3 > span')).click();

      break;

    case "Men":

      await driver.findElement(By.css('#ui-id-4 > span:nth-child(2)')).click();

      break;

    case "Women":

      await driver.findElement(By.css('#ui-id-5 > span:nth-child(2)')).click();

      break;

    case "Gear":

      await driver.findElement(By.css('#ui-id-6 > span:nth-child(2)')).click();

      break;

    case "Training":

      await driver.findElement(By.css('#ui-id-7 > span:nth-child(2)')).click();

      break;

    case "Sale":

      await driver.findElement(By.css('#ui-id-8 > span')).click();

      break;

    default:

      throw new Error(`Invalid menu item: ${menuItem}`);

  }

});

Then('I should see the {string} page', async function (expectedPageTitle) {

  const actualPageTitle = await driver.getTitle();

  assert.strictEqual(actualPageTitle, expectedPageTitle, `Expected page title "${expectedPageTitle}", but got "${actualPageTitle}"`);

});

After(async function() {

  await driver.quit()

})

### US1-TS2-Navigation-NP.feature

**Feature:** Navigation

As a user, I want to be able to navigate the Luma website easily and find the products I'm looking for quickly, so that I can make a

purchase without any hassle.

**Scenario**: Verify users cannot place a product in the basket without entering size and color (Negative path)

Given the user is on the home page of the Luma website

When they select the "Tops" category

And they select the "Breathe-Easy Tank" product

And they click the "Add to Cart" button

Then an error message is displayed indicating that the size and color must be selected

### **US1-TS2-Navigation-NP.js**

**const { Given, When, Then } = require('cucumber')**

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('I am on the home page', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('I navigate to the Tops category', async function() {

  await driver.findElement(By.linkText("Women")).click();

  await driver.findElement(By.linkText("Tops")).click();

});

When('I add the Breathe-Easy Tank to the cart', async function() {

  await driver.findElement(By.linkText("Breathe-Easy Tank")).click();

  await driver.findElement(By.css("#product-addtocart-button > span")).click();

});

Then('I should see the error message for the required size selection', async function() {

  const errorMessage = await driver.findElement(By.css(".message.error"));

  const errorMessageText = await errorMessage.getText();

  assert.strictEqual(errorMessageText, "You need to choose options for your item.");

});

After(async function() {

  await driver.quit()

})

### US2-TS1-Search-Functionality-HP.feature

**Feature:** Search Functionality

As a user, I want to be able to search for products on the Luma website and see accurate results, so that I can find the products I

want without having to browse through multiple pages.

**Scenario**: Test the search functionality to ensure accurate search results (happy path)

Given the user is on the home page of the Luma website

When they enter product name in the search bar

And they select the first search suggestion

Then they should see accurate search results

### US2-TS1-Search-Functionality-HP.js

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('they enter "didi" in the search bar', async function () {

  await driver.findElement(By.id('search')).click();

  await driver.findElement(By.id('search')).sendKeys('didi', Key.RETURN);

});

Then('they should see accurate search results', async function () {

  let resultTitle = await driver.findElement(By.css('.page-title span')).getText();

  assert.strictEqual(resultTitle, 'Didi Sport Watch');

  let firstResult = await driver.findElement(By.css('.product-item:first-child .product-item-name a')).getText();

  assert.strictEqual(firstResult, 'Didi Sport Watch');

});

After(async function () {

  await driver.quit();

});

### US2-TS2-Search-Functionality-NP.feature

**Feature:** Search functionality

As a user, I want to be able to search for products on the Luma website and see accurate results, so that I can find the products I

want without having to browse through multiple pages.

**Scenario**: Prevent navigation to non-existing product (negative path)

Given the user is on the Luma website

When the user enters a non-existing product name in the search bar

And clicks on the search button

Then they should not be navigated to any product page

And they should be redirected to a separate page with an error message indicating that no matching products were found

### US2-TS2-Search-Functionality-NP.js

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('the user enters a non-existing "PRODUCT\_NAME" in the search bar', async function () {

  await driver.findElement(By.id('search')).click();

  await driver.findElement(By.id('search')).sendKeys('didia', Key.RETURN);

});

Then('they should not be navigated to any product page', async function () {

  let currentUrl = await driver.getCurrentUrl();

  assert.strictEqual(currentUrl, 'https://magento.softwaretestingboard.com/catalogsearch/result/?q=didia');

});

And('they should be redirected to a separate page with an error message indicating that no matching products were found', async function () {

  let errorMessage = await driver.findElement(By.css('.page.messages > div')).getText();

  assert.strictEqual(errorMessage, 'Your search returned no results.');

});

After(async function () {

  await driver.quit();

});

### US3-TS1-Product-Listing-and-Details-HP.feature

**Feature**: Product listing and details

As a user, I want to be able to view accurate product details such as prices, descriptions, and images, so that I can make an informed

decision when purchasing products.

**Scenario**: Test that users can view more details for available products (happy path)

Given the user is on the home page of the Luma website

When they click on the "Gear" menu

And they click on the "Watches" menu

And they click on the "Didi Sport Watch" product

And they click on the "Details" tab

Then they should be able to view information in "Details" tab

### US3-TS1-Product-Listing-and-Details-HP.js

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given(/^the user is on the Luma website$/, async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When(/^they click on the "Gear" menu$/, async function () {

  await driver.findElement(By.css("#ui-id-6 > span:nth-child(2)")).click();

});

And(/^they click on the "Watches" link$/, async function () {

  await driver.findElement(By.css("#ui-id-27 > span:nth-child(2)")).click();

});

And(/^they click on the "Didi Sport Watch" link$/, async function () {

  await driver.findElement(By.css(".items:nth-child(2) > .item:nth-child(3) > a")).click();

});

And(/^they click on the "Details" tab$/, async function () {

  await driver.findElement(By.id("tab-label-additional-title")).click();

});

Then(/^they should see the description of the Didi Sport Watch$/, async function () {

  // Verify that the description of the Didi Sport Watch is displayed on the page

  const description = await driver.findElement(By.css('.product.attribute.description .value'));

  const expectedText = "The Didi Sport Watch helps you keep your workout plan down to the second. The vertical, digital face looks sleek and futuristic. This watch is programmed with tons of helpful features such as a timer, an alarm clock, a pedometer, and more to help make your exercise more productive.";

  assert(await description.getText() === expectedText, "Description of the Didi Sport Watch is not displayed");

});

After(async function () {

  await driver.quit();

})

### **US3-TS**2**-Product-Listing-and-Details-**N**P.**feature

**Feature:** Product listing and details

As a user, I want to be able to view accurate product details such as prices, descriptions, and images, so that I can make an informed

decision when purchasing products.

**Scenario**: Test that users cannot view related products in Sale menu if in the product category there are not products with discount

price

Given the user is on the home page of the Luma website

When they click on the "Men" menu

And they click on the "Bottoms" menu

And they click on the "Directory" filter

And they click on the "Pants" menu

And they click the "Sale" filter

Then they should not see any related products in Sale

### US3-TS2-Product-Listing-and-Details-NP.js

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

const assert = require('assert');

let driver;

Given(/^the user is on the Luma website$/, async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get('https://magento.softwaretestingboard.com/');

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When(/^they click on the "Men" menu$/, async function () {

  await driver.findElement(By.css("#ui-id-5 > span:nth-child(2)")).click()

})

When(/^they click on the "Bottoms" link$/, async function () {

  await driver.findElement(By.linkText("Bottoms")).click();

});

When(/^they click on the "Directory" filter$/, async function () {

  await driver.findElement(By.css(".filter-options-item:nth-child(1) > .filter-options-title")).click();

});

When(/^they click on the "Pants" menu$/, async function () {

  await driver.findElement(By.css(".allow .item:nth-child(1) > a")).click();

});

When(/^they click on the "Sale" filter$/, async function () {

  await driver.findElement(By.css(".filter-options-item:nth-child(10) > .filter-options-title")).click();

});

Then(/^they should see that no related products are displayed in the Sale menu$/, async function () {

  const relatedProducts = await driver.findElements(By.css(".block-related .products-grid li"));

  assert.strictEqual(relatedProducts.length, 0, "Expected no related products to be displayed in the Sale menu");

});

After(async function() {

  await driver.quit();

});

### US4-TS1-Add-to-Cart-and-Checkout-HP.feature

**Feature:** Add to cart and checkout

As a user, I want to be able to add products to my cart and checkout quickly and easily, so that I can complete my purchase without

any delays.

**Scenario**: Test adding products to the cart and verify that the correct products and quantities are added (happy path).

Given the user is on the home page of the Luma website

When they click on the "Gear" menu

And they click on the "Watches" menu

And they click on the "Didi Sport Watch" product

And they add a product to the cart

And they navigate to the cart

And they proceed to checkout

Then they should see the product in the basket case

### US4-TS1-Add-to-Cart-and-Checkout-HP.js

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the home page of the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get("https://magento.softwaretestingboard.com/");

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When(/^they click on the "Gear" menu$/, async function () {

  await driver.findElement(By.css("#ui-id-6 > span:nth-child(2)")).click();

});

When(/^they click on the "Watches" menu$/, async function () {

  await driver.findElement(By.linkText("Watches")).click();

});

When(/^they click on the "Didi Sport Watch" product$/, async function () {

  await driver.findElement(By.linkText("Didi Sport Watch")).click();

});

When('they add a product to the cart', async function () {

  await driver.findElement(By.css("#product-addtocart-button > span")).click();

});

When('they navigate to the cart', async function () {

  await driver.findElement(By.css(".showcart")).click();

});

When('they proceed to checkout', async function () {

  await driver.findElement(By.id("top-cart-btn-checkout")).click();

  await driver.findElement(By.css("strong > span:nth-child(2)")).click();

  await driver.findElement(By.css(".opc-block-summary")).click();

});

Then('they should see the product in the basket case', async function () {

// Verify that the product was added to the cart and visible on the checkout page

  const productName = await driver.findElement(By.css(".product-item-name > a")).getText();

  assert.strictEqual(productName, "Didi Sport Watch", "Product name did not match expected value");

  const numItems = await driver.findElement(By.css(".counter-number")).getText();

  assert.strictEqual(numItems, "1", "Number of items in cart did not match expected value");

});

After(async function () {

  await driver.quit();

})

### **US4-TS2-Add-to-Cart-and-Checkout-NP.feature**

**Feature**: Add to cart and checkout

As a user, I want to be able to add products to my cart and checkout quickly and easily, so that I can complete my purchase without

any delays.

**Scenario**: Test subtract all products from the cart and verify that couldn’t checkout with empty basket (negative path).

Given the user is on the home page of the Luma website

When they click on the "Gear" menu

And they click on the "Watches" menu

And they click on the "Didi Sport Watch" product

And they add a product to the cart

And they click on the "View cart" button

And they click on the "Remove item" link

Then they should not be able to checkout with an empty basket

### **US4-TS2-Add-to-Cart-and-Checkout-NP.js**

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the home page of the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get("https://magento.softwaretestingboard.com/");

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When(/^they click on the "Gear" menu$/, async function () {

  await driver.findElement(By.css("#ui-id-6 > span:nth-child(2)")).click();

});

When(/^they click on the "Watches" menu$/, async function () {

  await driver.findElement(By.linkText("Watches")).click();

});

When(/^they click on the "Didi Sport Watch" product$/, async function () {

  await driver.findElement(By.linkText("Didi Sport Watch")).click();

});

When('they add a product to the cart', async function () {

  await driver.findElement(By.css("#product-addtocart-button > span")).click();

});

When('they navigate to the cart', async function () {

  await driver.findElement(By.css(".showcart")).click();

});

When(/^they click on the cart quantity$/, async function () {

  await driver.findElement(By.css(".counter-number")).click();

});

When(/^they click on the "Remove" button$/, async function () {

  await driver.findElement(By.linkText("Remove")).click();

});

Then(/^I should not be able to checkout with an empty basket$/, async function () {

  const emptyCartMessage = await driver.findElement(By.css(".cart-empty"));

  const isDisplayed = await emptyCartMessage.isDisplayed();

  assert.ok(isDisplayed, "Expected empty cart message to be displayed, but it was not");

});

After(async function () {

  await driver.quit();

})

### US5-TS1-Account-Creation-and-Login-HP.feature

**Feature**: Account creation and login

As a user, I want to be able to create an account and log in to the Luma website easily, so that I can view my order history and make

future purchases more conveniently.

**Scenario**: Test the account creation process and verify that users can create accounts (happy path).

Given the user is on the home page of the Luma website

When they click on "Create an Account" link

And they fill inputs in the registration form:

"First Name", "Last Name", "Email", "Password", "Confirm Password"

And they submit the form with click on "Create and Account" button

Then they should be redirected to the home page of the registered customer

### US5-TS1-Account-Creation-and-Login-HP.js

const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the home page of the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get("https://magento.softwaretestingboard.com/");

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('I click on "Create an Account" link', async function () {

  await driver.findElement(By.css("ul.header:nth-child(2) > li:nth-child(3) > a:nth-child(1)")).click();

  await driver.wait(until.urlIs("https://magento.softwaretestingboard.com/customer/account/create/"), 5000);

});

When('I enter the last name {string}', async function (lastName) {

  await driver.findElement(By.id("lastname")).click();

  await driver.findElement(By.id("lastname")).sendKeys(lastName);

});

When('I enter the email address {string}', async function (emailAddress) {

  await driver.findElement(By.id("email\_address")).click();

  await driver.findElement(By.id("email\_address")).sendKeys(emailAddress);

});

When('I enter the password {string}', async function (password) {

  await driver.findElement(By.id("password")).click();

  await driver.findElement(By.id("password")).sendKeys(password);

});

When('I confirm the password {string}', async function (passwordConfirmation) {

  await driver.findElement(By.id("password-confirmation")).click();

  await driver.findElement(By.id("password-confirmation")).sendKeys(passwordConfirmation);

});

When('I click on the "Create an Account" button', async function () {

  await driver.findElement(By.css(".submit > span")).click();

});

Then('I should be redirected to the welcome page for the new registered customer', async function () {

  const currentUrl = await driver.getCurrentUrl();

  assert.strictEqual(currentUrl, "https://magento.softwaretestingboard.com/customer/account/");

});

After(async function() {

  await driver.quit();

});

### **US5-TS**2**-Account-Creation-and-Login-**N**P.**feature

**Feature**: Account creation and login

As a user, I want to be able to create an account and log in to the Luma website easily, so that I can view my order history and make

future purchases more conveniently.

**Scenario**: Test that users can not create accounts when all necessary inputs were not done (negative path).

Given the user is on the home page of the Luma website

When they click on "Create an Account" link

And they fill inputs in the registration form:

"First Name", "Last Name", "Email", "Password"

And they submit the form with click on "Create and Account" button

Then they should not be redirected to the home page of the registered customer

### **US5-TS2-Account-Creation-and-Login-NP.js** const { Given, When, Then } = require('cucumber');

const { Builder, By, Key, until } = require('selenium-webdriver');

const assert = require('assert');

let driver;

Given('the user is on the home page of the Luma website', async function () {

  driver = await new Builder().forBrowser('firefox').build();

  await driver.get("https://magento.softwaretestingboard.com/");

  await driver.manage().window().setRect({ width: 1721, height: 927 });

});

When('I click on "Create an Account" link', async function () {

  await driver.findElement(By.css("ul.header:nth-child(2) > li:nth-child(3) > a:nth-child(1)")).click();

  await driver.wait(until.urlIs("https://magento.softwaretestingboard.com/customer/account/create/"), 5000);

});

When('I enter the last name {string}', async function (lastName) {

  await driver.findElement(By.id("lastname")).click();

  await driver.findElement(By.id("lastname")).sendKeys(lastName);

});

When('I enter the email address {string}', async function (emailAddress) {

  await driver.findElement(By.id("email\_address")).click();

  await driver.findElement(By.id("email\_address")).sendKeys(emailAddress);

});

When('I enter the password {string}', async function (password) {

  await driver.findElement(By.id("password")).click();

  await driver.findElement(By.id("password")).sendKeys(password);

});

When('I click on the "Create an Account" button', async function () {

  await driver.findElement(By.css(".submit > span")).click();

});

Then('I should not be redirected to the welcome page for the new registered customer', async function () {

  const currentUrl = await driver.getCurrentUrl();

  assert.notStrictEqual(currentUrl, "https://magento.softwaretestingboard.com/customer/account/");

});

After(async function() {

  await driver.quit();

});