# Automation Testing Project

# NopCommerce

(*Selenium* + *TestNG*)

---

## 📜 Description

Automate the following **functionalities** on the *NopCommerce* demo website (https://demo.nopcommerce.com/) using ***Selenium WebDriver***, ***TestNG***, and ***Page Object Model*** **(*POM*)**.

---

## ✅ Scenario #1: Registration

**Test Case: Valid Register**
User can register a new account successfully.

**Steps:**

1. Navigate to the Register Page.
2. Select gender type.
3. Enter first name = automation and last name = tester.
4. Enter Date of Birth: Day, Month, Year.
5. Enter email = test@example.com.
6. Enter and confirm password = P@ssw0rd.
7. Click Register.
8. Assert:
    - Registration success message is displayed.
    - Message text = "Your registration completed"
    - Message color = rgba(76, 177, 124, 1)

## ✅ Scenario #2: Login

### Test Case #1: Valid Login

**Steps:**

1. Navigate to the Login page.
2. Enter a valid email: test@example.com
   and password: P@ssw0rd
3. Click the Login button.
4. Assert the following using **SoftAssert**:
   - The current URL is https://demo.nopcommerce.com/
   - The "My account" tab is displayed

Call assertAll() at the end.

---

### Test Case #2: Invalid Login

**Steps:**

1. Navigate to the Login page.
2. Enter an invalid email: wrong@example.com
   and password: P@ssw0rd
3. Click the Login button.
4. Assert the following using **SoftAssert**:
   - The error message contains "Login was unsuccessful."
   - The color of the error message is red (Hex: #e4434b)

⚠️ Note: The output color from getCssValue("color") is in RGBA format.
Convert it to Hex using the following utility:

```
import org.openqa.selenium.support.Color;

Color.fromString("rgba(228, 67, 75, 1)").asHex();
```

## ✅ Scenario #3: Currencies

**Test Case: Verify Euro Symbol (€) Appears for All Products**

**Steps:**

1. From the **currency dropdown list** at the top left of the homepage, select **"Euro"**.
2. Use **hard assertions** to verify that the **Euro symbol (€)** is displayed for **each of the 4 products shown on the homepage**.

✳ Details for Step 2:

- Use driver.findElements() to capture all product price elements on the homepage.
- Iterate through the list using a for loop:
    - Use .get(i).getText() to get each product price as a string.
    - Store the value in a variable.
    - Use Assert.assertTrue(variable.contains("€")) to verify the symbol is present.

## ✅ Scenario #4: Search

**Test Case #1: Search Using Product Name**

Implement a TestNG test that loops over a set of product names:
**book, laptop, nike**

**Steps:**

1. Enter each product name into the search input box and click the Search button.
2. Use **soft assertions** to verify the following for each product:
    - The current URL contains: https://demo.nopcommerce.com/search?q=
    - The search results section is populated:
        - Use findElements() and size() to count the number of search results.
        - Loop over each result and use .getText().toLowerCase() to verify it contains the search word.

Call assertAll() at the end of each search execution.

---

**Test Case #2: Search Using Product SKU**

Loop over a list of SKUs:
**SCI_FAITH, APPLE_CAM, SF_PRO_11**

SKU = product serial number (example: AP_MBP_13 for MacBook Pro)

**Steps:**

1. Enter each SKU into the search input box and click the Search button.
2. Click on the product link from the search results.
3. On the product details page, use **hard assertions** to verify:
    - The SKU shown on the product page **contains the searched SKU**.

You should pass each SKU dynamically and assert it accurately.

## ✅ Scenario #5: Home Sliders

**Test Case #1: Validate Nokia Lumia 1020 Slider**

**Steps:**

1. Wait for the **first slider** to become visible and clickable.
2. Click the first slider.
3. Wait for the page to load using **explicit wait** and check the current URL.
4. Use a **hard assertion** to verify that the URL is equal to: https://demo.nopcommerce.com/nokia-lumia-1020

⚠️ **Note**: The actual result is that the URL remains https://demo.nopcommerce.com/, which is incorrect.
This means the test should **fail** and be considered a valid **UI bug** detection.

---

**Test Case #2: Validate iPhone 6 Slider**

**Steps:**

1. Wait for the **second slider** to become visible and clickable.
2. Click the second slider.
3. Wait for the page to load using **explicit wait** and check the current URL.
4. Use a **hard assertion** to verify that the URL is equal to: https://demo.nopcommerce.com/iphone-6

⚠️ Again, the actual behavior keeps the user at https://demo.nopcommerce.com/, so the automation should **fail** this test as well and flag it as a **defect**.

---

## ✓ Scenario #6: Follow Us

**Test Case #1: Verify Facebook link**

- Click on the Facebook icon.
- Wait for the second tab to open.
- Switch to the new tab.
- Assert the URL is https://www.facebook.com/nopCommerce.

**Test Case #2: Verify Twitter link**

- Click on the Twitter icon.
- Assert the new tab opens with https://twitter.com/nopCommerce.

**Test Case #3: Verify RSS link**

- Click on the RSS icon.
- Expected URL: https://demo.nopcommerce.com/new-online-store-is-open.
- **Note:** This test should **fail**, as the actual URL is https://demo.nopcommerce.com/news/rss/1.

**Test Case #4: Verify YouTube link**

- Click on the YouTube icon.
- Assert the new tab URL is https://www.youtube.com/user/nopCommerce.

## ✅ Scenario #7: Wishlist

**Test Case #1: Add item to wishlist and verify success message**

1. Open the website: https://demo.nopcommerce.com/
2. Locate the product: **HTC One M8 Android L 5.0 Lollipop** (it is one of the 4 products in the center of the home page).
3. Click on the ❤️ (wishlist) button under that product.
4. After clicking, a green success message should appear:

"The product has been added to your wishlist"

5. Use **soft assertions** to validate:
   - The **success message is displayed**
   - The **background color** of the message is **green**

💡 *Tip:* If you're validating color in HEX format, you may need to **convert RGBA to HEX** using utility functions.

---

**Test Case #2: Validate item appears in the wishlist**

1. On the same website, click the ❤️ button again for the same product.
2. Wait until the success message disappears (use **explicit wait** with ExpectedConditions.invisibilityOfElement...).
3. Then, click the **Wishlist** tab at the top right (beside "Register", "Log in").
4. From the Wishlist page:
   - Get the **Qty** value for the item added.
   - Assert that it is **greater than 0** (indicating the item was successfully added).

💡 *Tip:* Decide if you will use:

- getText() to extract the number
- Or getAttribute("value") if it's in an input box

- Or getCssValue() if the value is set via style