AUTOMATION PROJECT CREATED WITH SELENIUM WEBDRIVER AND PYTHON + UNITTEST

Author:

Katarzyna Sycz, grupa 2, WSB Wroclaw
30.04.2018

1. Test Cases

TC001

Title: Searching for an element in the main search module.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.
2. In the main search field, type the “dress” word.
3. Click the button with the magnifier icon.

Expected results:

- Website shows found products that contains the ‘dress’ word in the name.

TC002

Title: Opening a contact form.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.
2. Click the **Contact us** link in the top-right corner.

Expected results:

- The contact form should appear.

TC003

Title: Filling out all fields in the contact form and submitting it.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.

2. Click the **Contact us** link in the top-right corner.
3. Click the drop-down menu under **Subject Heading** and select the second option - **Webmaster**.
4. Type the valid **E-mail address** - test@test.com.
5. Type the **Order reference** - OR12345.
6. Type the **Message** - This is a testing message.
7. Click the **Send** button.

Expected results:

- The contact form should be submitted - the **Your message has been successfully sent to our team.** should appear.

TC004

Title: Submitting the contact form with an invalid e-mail address.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.
2. Click the **Contact us** link in the top-right corner.
3. Click the drop-down menu under **Subject Heading** and select the second option - **Webmaster.**
4. Type the invalid **E-mail address** - test.
5. Type the **Order reference** - OR12345.
6. Type the **Message** - This is a testing message.
7. Click the **Send** button.

Expected results:

- The contact form should not be submitted - the **Invalid email address.** message should appear.

TC005

Title: Opening a category tab.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.

2. Click the **T-SHIRTS** tab.

Expected results:

- After the website reloads, the category name should show **T-SHIRTS.**

TC006

Title: Adding an item to the cart.

Environment:

- Google Chrome - Version 66.0.3359.117 (Official Build) (64-bit)
- CentOS 7

Preconditions:

- User is not logged in.
- The Google Chrome browser is open.

Steps:

1. Open the http://automationpractice.com website.
2. Click the **T-SHIRTS** tab.
3. Click the name of the product that appears as the first one.
4. Click the **Add to cart** button.
5.

Expected results:

- The confirmation about adding the product to the cart appears - **Product successfully added to your shopping cart.**

2. Code:

```
import unittest
import time
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support.select import Select

class AutomationPractice(unittest.TestCase):
  def setUp(self):
     # create a new Chrome session
     self.driver = webdriver.Chrome()
     self.driver.maximize_window()
     # navigate to the home page
     self.driver.get("http://automationpractice.com")
  def test_main_search(self):
     self.searchInput = self.driver.find_element_by_id("search_query_top")
     self.searchInput.send_keys("dress")
     self.searchBtn = self.driver.find_element_by_name("submit_search")
     self.searchBtn.click()
     time.sleep(5)
     self.foundProduct =
self.driver.find_element_by_class_name("product-name").text
     self.assertIn(self.foundProduct, "dress")

  def test_contact_form_open(self):
     self.contactUs = self.driver.find_element_by_id('contact-link')
     self.contactUs.click()
     self.contactForm = self.driver.find_element_by_class_name('contact-form-box')
     self.assertTrue(self.contactForm)

  def test_contact_form_full(self):
     self.contactUs = self.driver.find_element_by_id('contact-link')
     self.contactUs.click()
     self.subject = self.driver.find_element_by_id('id_contact')
     self.select = Select(self.subject)
     self.select.select_by_value('2')
     self.email = self.driver.find_element_by_id('email')
     self.email.send_keys('test@test.com')
     self.order = self.driver.find_element_by_id('id_order')
     self.order.send_keys("OR12345")
     self.message = self.driver.find_element_by_id('message')
     self.message.send_keys("This is a testing message")
     self.sendBtn = self.driver.find_element_by_id('submitMessage')
     self.sendBtn.click()
     self.successMessage =
self.driver.find_element_by_class_name('alert-success').text
     self.assertEqual(self.successMessage, 'Your message has been successfully
sent to our team.')
```

```
    def test_contact_form_incorrect_email(self):
        self.contactUs = self.driver.find_element_by_id('contact-link')
        self.contactUs.click()
        self.subject = self.driver.find_element_by_id('id_contact')
        self.select = Select(self.subject)
        self.select.select_by_value('2')
        self.email = self.driver.find_element_by_id('email')
        self.email.send_keys('test')
        self.order = self.driver.find_element_by_id('id_order')
        self.order.send_keys("OR12345")
        self.message = self.driver.find_element_by_id('message')
        self.message.send_keys("This is a testing message")
        self.sendBtn = self.driver.find_element_by_id('submitMessage')
        self.sendBtn.click()
        self.errorMessage = self.driver.find_element_by_css_selector('#center_column
> div > ol > li').text
        self.assertEqual(self.errorMessage, 'Invalid email address.')

    def test_shirt_tab_open(self):
        self.tShirtTab = self.driver.find_element_by_css_selector('#block_top_menu > ul
> li:nth-child(3)')
        self.tShirtTab.click()
        self.productCatName = self.driver.find_element_by_class_name('cat-name').text
        self.assertEqual(self.productCatName, 'T-SHIRTS ')

    def test_cart_add_item(self):
        self.tShirtTab = self.driver.find_element_by_css_selector('#block_top_menu > ul
> li:nth-child(3)')
        self.tShirtTab.click()
        self.productSelect = self.driver.find_element_by_css_selector('#center_column >
ul > li > div > div.right-block > h5 > a')
        self.productSelect.click()
        self.addToCartBtn = self.driver.find_element_by_css_selector('#add_to_cart >
button')
```

```
        self.addToCartBtn.click()
        time.sleep(5)
        self.addedItemMessage = self.driver.find_element_by_css_selector('#layer_cart
> div > div.layer_cart_product > h2').text
        self.assertEqual(self.addedItemMessage, 'Product successfully added to your
shopping cart')

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main(verbosity = 2)
```