Automatyzacja przypadku testowego przy pomocy Selenium Webdriver



Autor

Krzysztof Gałuszka

Poznań 2022

1. **Przypadek testowy**

ID. 001

**Tytuł:** Rejestracja nowego użytkownika na platformie Sanpolprofi.pl, bez podania imienia w formularzu rejestracyjnym.

**Środowisko:** Chrome wersja 94.9.4606.61 (64-bitowa), system operacyjny Windows 10

**Warunek wstępny**: Otwarta strona główna. Użytkownik niezalogowany

**Kroki:**

1. Klikamy „Rejestracja”
2. Wpisz e-mail
3. Wpisz nazwisko
4. Wpisz numer telefonu
5. Wpisz pełną nazwę firmy
6. Wpisz ulice
7. Wpisz numer ulicy
8. Wpisz kod pocztowy
9. Wpisz miejscowość
10. Wpisz NIP
11. Zaakceptuj regulamin i politykę prywatności
12. Kliknij przycisk „PRZEŚLIJ DANE”

**Oczekiwany rezultat:**

1. Użytkownik nie zarejestrował się do systemu
2. Użytkownik otrzymuję informację, że pole jest wymagane
3. **Automatyzacja przypadku testowego przy pomocy Selenium Webdriver**

import unittest

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

import time

# Dane tesowe

name\_1 = "Krzysztof"

surname = "Testowy"

account\_Name = "testowyuzytkownik@wp.pl"

password ="Test01."

Phone\_number = "111222333"

Company\_Name = "Testoweanie"

Street\_1 = "Wolczynskiego"

Street\_number = "6"

Post\_code\_1 = "51-630"

City\_1 = "Poznan"

Nip\_1 = "423423423"

class AutomationRegistration(unittest.TestCase):

    def setUp(self):

        # Przygotowanie testu

        self.driver = webdriver.Chrome()

        self.driver.get("http://sanpolprofi.pl/customer/b2b/login/")

    def testRegister(self):

        driver = self.driver

        # Kliknij przycisk Rejestracja

        register\_button = driver.find\_element(By.CLASS\_NAME, "login\_\_link--secondary")

        register\_button.click()

        # Wpisz Login do pola Login

        Insert\_login = driver.find\_element(By.ID, "b2b-email")

        Insert\_login.send\_keys(account\_Name)

        # Wpisz nazwisko do pola Nazwisko

        Insert\_Surname = driver.find\_element(By.ID, "b2b-last-name")

        Insert\_Surname.send\_keys(surname)

        # Wpisz numer telefonu do pola Nr telefonu

        Insert\_Phone = driver.find\_element(By.ID, "b2b-phone")

        Insert\_Phone.send\_keys(Phone\_number)

        # Wpisz nazwę firmy do pola Pełna nazwa firmy

        Insert\_Company = driver.find\_element(By.ID, "b2b-company-name")

        Insert\_Company.send\_keys(Company\_Name)

        # Wpisz ulice do pola Ulica

        Insert\_Street = driver.find\_element(By.ID, "b2b-street")

        Insert\_Street.send\_keys(Street\_1)

        # Wpisz numer ulicy do pola Numer

        Insert\_Street\_number = driver.find\_element(By.XPATH, "/html/body/div[2]/main/div[3]/div/section/form/fieldset[2]/div/div[2]/div[2]/div/input")

        Insert\_Street\_number.send\_keys(Street\_number)

        # Wpisz kod pocztowy do pola Kod pocztowy

        Insert\_Post\_Code = driver.find\_element(By.ID, "b2b-zip-code")

        Insert\_Post\_Code.send\_keys(Post\_code\_1)

        # Wpisz miejscowosc do pola Miejscowość

        Insert\_City = driver.find\_element(By.ID, "b2b-city")

        Insert\_City.send\_keys(City\_1)

        # Wpisz NIP do pola NIP

        Insert\_Nip = driver.find\_element(By.ID, "b2b-vat-number")

        Insert\_Nip.send\_keys(Nip\_1)

        # Zaznacz checkoboxa "Oświadczam, że zapoznałem się z Regulaminem oraz Polityką Prywatności i akceptuję ich warunki"

        Insert\_checkbox = driver.find\_element(By.CLASS\_NAME, "modal-trigger")

        Insert\_checkbox.click()

        # Odczekaj 2 sekundy aż wyświetli się regulamin

        time.sleep(2)

        # Spawdź czy pole "AKCEPTUJĘ REGULAMIN" wyświetliło się na stronie, jeśli nie to poczekaj 10 sekund to tego czasu, jak wyświeliło się to kliknij w przycisk

        try:

            Accept\_Checkobox  = WebDriverWait(driver, 10).until(

            EC.presence\_of\_element\_located((By.XPATH, "/html/body/div[2]/main/div[3]/div/div[1]/div/section/div/button[3]"))

            )

            Accept\_Checkobox .click()

        except:

            print("")

        # Kliknij przycisk "PRZEŚLIJ DANE" w celu zarejestrowania się do portalu

        Register\_button = driver.find\_element(By.XPATH, "/html/body/div[2]/main/div[3]/div/section/form/button[2]")

        Register\_button.click()

        # Ustalam 3 zmienne, które będę wykorzystywać do znalezienia i sprawdzenia błędów

        i=0

        x=0

        tab = []

        #Wyszukuje sekcję na której znajduje się formularz

        error\_all = driver.find\_element(By.XPATH, "/html/body/div[2]/main/div[3]/div/section")

        #Wyszukuje wszyskie elementy które są divem w formularzu

        errors\_search = error\_all.find\_elements(By.TAG\_NAME, "div")

        #Tworzę pętle, która wypisuje wszystkie komunikaty w formularzu

        for error in errors\_search:

            #Każdy komunikat dopisuję do tablicy

            tab.append(error.text)

            #Każdy błąd ma komunikat "To jest wymagane pole", więc jeśli bląd wystąpi to używam instrukcji warunkowej if

            if(error.text == "To jest wymagane pole."):

                # Zmienna i zlicza ilość wystąpienia błędów na stronie

                i=i+1

                # Przypisuje zmiennej s wartość komunikatu o błędzie, z racji, że komunikat ma formę (nazwa pola + komunikat o błędzie), to rozdzielam

                # całość, na dwie tablicę. Używam poprzedniego elementu tablicy, gdyż komunikat o błędzie jest aktualnym elementem tablicy t[x], a poprzedni element,

                # to miejsce w którym wystąpił błąd.

                s=tab[x-1].split("\n")

                # Jeśli błąd pojawił się w innym miejscu niż przy wpisaniu imienia, to funckja zwraca nam błąd

                self.assertEqual("IMIĘ", s[0], "Error message is not correct")

            x=x+1

        # Jeśli istnieje więcej niż jedno miejsce w którym występuje błąd to funkcja zwraca błąd i podaje ile błędów występuje w formularzu

        self.assertEqual(1,  i,"\nThere are more errors than 1 \nNumbers of errors: " + str(i))

        # Zamknięcie testu

        def tearDown(self):

            self.driver.close()

if \_\_name\_\_ == "\_\_main\_\_":

    #Inicjuje wszystkie unitesty

    unittest.main()

1. Uwagi końcowe

Automatyzacja przypadku testowego powiodła się. Użytkownik nie został zarejestrowany do platformy, gdyż popełnił błąd w formularzu rejestracyjnym. Test może być wrażliwy na zmianę struktury strony , gdyż konieczne było zastosowanie długich ścieżek w lokalizatorach XPATH.