Санкт-Петербургский Национальный Исследовательский
Университет Информационных Технологий, Механики и Оптики

Факультет «Инфокоммуникационных Технологий»
Направление подготовки «Программирование в
инфокоммуникационных системах»

Лабораторная работа №3

Выполнил:
Крылов Дан Станиславович
Группа №3322
Проверил:
Кочубеев Николай Сергеевич

Санкт-Петербург
2024

Цель работы.

протестировать работу открытого кода из GitHub с помощью E2E-тестов.

Задачи.

1. Выбор репозитория с GitHub
2. Анализ тестируемых функциональностей
3. Написание E2E тестов

Ход работы.

1. **Выбор репозитория.**

   Был выбран репозиторий с GitHub, созданный сайт находится по ссылке: https://bookcart.azurewebsites.net/

   Он представляет собой сайт онлайн магазина книг, можно добавить книги в корзину, можно зарегистрироваться, войти в личный кабинет.

2. **Анализ тестируемых функциональностей**

   Важные случаи использования (Use Cases):

   • Добавление товара в корзину зарегистрированным пользователем

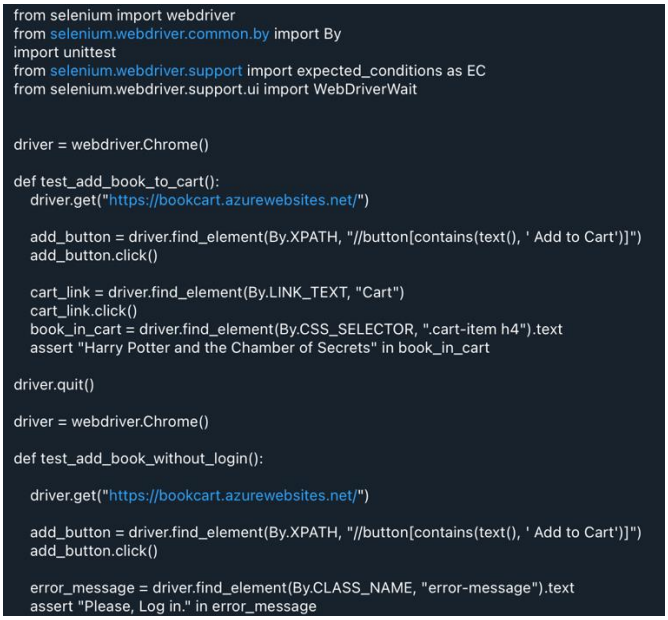
   • Добавление товара в корзину незарегистрированным пользователем

   • Вход в систему с правильными данными

   • Вход в систему с неправильными данными

3. **Написание тестов.**

   На рисунке 1 код тестов, которые проверяют, добавление товара в корзину авторизованным пользователем (добавление в корзину должно сработать) и неавторизованным пользователем (добавление в корзину не должно сработать, должно появиться сообщение об ошибке).



```
from selenium import webdriver
from selenium.webdriver.common.by import By
import unittest
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import WebDriverWait


driver = webdriver.Chrome()

def test_add_book_to_cart():
    driver.get("https://bookcart.azurewebsites.net/")

    add_button = driver.find_element(By.XPATH, "//button[contains(text(), ' Add to Cart')]")
    add_button.click()

    cart_link = driver.find_element(By.LINK_TEXT, "Cart")
    cart_link.click()
    book_in_cart = driver.find_element(By.CSS_SELECTOR, ".cart-item h4").text
    assert "Harry Potter and the Chamber of Secrets" in book_in_cart

driver.quit()

driver = webdriver.Chrome()

def test_add_book_without_login():

    driver.get("https://bookcart.azurewebsites.net/")

    add_button = driver.find_element(By.XPATH, "//button[contains(text(), ' Add to Cart')]")
    add_button.click()

    error_message = driver.find_element(By.CLASS_NAME, "error-message").text
    assert "Please, Log in." in error_message
```

Рисунок 1 – Добавление товара.

На рисунке 2 код, который проверяет вход в систему с правильными данными и вход в систему с неправильными данными.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import unittest
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import WebDriverWait

class TestBookstoreLogin(unittest.TestCase):

    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.get("https://bookcart.azurewebsites.net/login")

    def test_successful_login(self):
        WebDriverWait(self.driver, 10).until(EC.presence_of_element_located((By.CSS_SELECTOR,
"[formControlName='username']")))

        username_input = self.driver.find_element(By.CSS_SELECTOR, "[formControlName='username']")
        password_input = self.driver.find_element(By.CSS_SELECTOR, "[formControlName='password']")
        login_button = self.driver.find_element(By.CSS_SELECTOR, "[class= 'mdc-button mdc-button--raised
mat-mdc-raised-button mat-primary mat-mdc-button-base']")
        username_input.clear()
        username_input.send_keys("testtesttest")
        password_input.clear()
        password_input.send_keys("Testtest1")
        login_button.click()
        button = self.driver.find_element(By.CSS_SELECTOR, "[class= 'mdc-button__label']").text
        self.assertEqual(button, "Book Cart", "Username or Password is incorrect.")

    def test_invalid_credentials(self):

        WebDriverWait(self.driver, 10).until(EC.presence_of_element_located((By.CSS_SELECTOR,
"[formControlName='username']")))

        username_input = self.driver.find_element(By.CSS_SELECTOR, "[formControlName='username']")
        password_input = self.driver.find_element(By.CSS_SELECTOR, "[formControlName='password']")
        login_button = self.driver.find_element(By.CSS_SELECTOR, "[class= 'mdc-button mdc-button--raised
mat-mdc-raised-button mat-primary mat-mdc-button-base']")
        username_input.clear()
        username_input.send_keys("123")
        password_input.clear()
        password_input.send_keys("123")
        login_button.click()
        button = self.driver.find_element(By.CSS_SELECTOR, "[class= 'mdc-button__label']").text
        self.assertEqual(button, "Book Cart", "Username or Password is incorrect.")

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

if name == "main":
    unittest.main()
```

Рисунок 2 – Авторизация.

Вывод:

Все тесты успешно пройдены. Достигнута цель работы - протестирована работа кода из GitHub с помощью E2E-тестов.