Warszawa, 23 maja 2022

STUDIA PODYPLOMOWE – TESTER AUTOMATYZUJĄCY W SELENIUM

AKADEMIA
LEONA KOŹMIŃSKIEGO

PROJEKT: AUTOMATYZACJA TESTÓW APLIKACJI WEBOWEJ W JAVA Z WYKORZYSTANIEM BIBLIOTEKI WEBDRIVER

Adres strony internetowej testowanego projektu:

http://sampleapp.tricentis.com/101/

[ ] Zastosowanie Data Driven Testing

[ ] Zastosowanie wzorca Page Object

[ ] Zastosowanie podejścia Page Factory

Autor/Autorzy:

Zuzanna Pawlak

Ocena: ……

1. **Opis projektu**

Przykład: Celem projektu jest przetestowanie formularza do wykupienia ubezpieczenia motoru, działania pola wyszukiwania oraz sprawdzenie poprawności hiperłączy. Wtym celu opracowano 7 przypadków testowych:

TC1: Poprawne uzupełnienie formularza do kupna ubezpieczenia motoru

TC2: Niepoprawne uzupełnienie uzupełnienia formularza do kupna ubezpieczenia motoru – podana niepoprawna pojemność silnika

TC3: Niepoprawne uzupełnienie uzupełnienia formularza do kupna ubezpieczenia motoru – podana niepoprawna moc silnika

TC4: Sprawdzenie poprawności zawartości listy rozwijanej

TC5: Sprawdzenie ilości hiperłączy na stronie głównej

TC6: Sprawdzenie poprawności wyświetlenia się hiperłączy na stronie głównej

TC7: Sprawdzenie poprawności działania pola wyszukiwania na stronie głównej

W niniejszej pracy zastosowano wzorzec projektowy Page Object, podejście Page Factory oraz testowanie oparte na danych – pobieranie danych z pliku Excel.

2. Przypadki testowe

Ze względu na ich złożoność przypadki testowe:

TC1: Poprawne uzupełnienie formularza do kupna ubezpieczenia motoru

TC2: Niepoprawne uzupełnienie uzupełnienia formularza do kupna ubezpieczenia motoru – podana niepoprawna pojemność silnika

TC3: Niepoprawne uzupełnienie uzupełnienia formularza do kupna ubezpieczenia motoru – podana niepoprawna moc silnika

TC4: Sprawdzenie poprawności zawartości listy rozwijanej

zostały opisane w pliku Excel, który zostanie dołączony do opisu projektu.

TC5: Sprawdzenie ilości hiperłączy na stronie głównej

Warunki wstępne:
Wchodzimy na stronę główną projektu
http://sampleapp.tricentis.com/101/ .
Kroki testowe :
Pobieramy wszystkie hiperłącza na stronie za pomocą wyszukiwania tagu <a>.
Oczekiwany rezultat :
Ilość hiperłączy to na stronie to 32. Za pomocą metody Assert.assertEquals sprawdzamy czy oczekiwana ilość hiperłączy jest równa z aktualną, pobraną ze strony.
Warunki końcowe :
Metoda Assert.assertEquals zwraca wartość True.

TC6: Sprawdzenie poprawności wyświetlenia się hiperłączy na stronie głównej

Warunki wstępne :
Wchodzimy na stronę główną projektu
http://sampleapp.tricentis.com/101/index.php
Kroki testowe
Pobieramy wszystkie hiperłącza na stronie za pomocą wyszukiwania tagu <a>. Za pomocą metody iterator przejdziemy po liście hiperłączy i dokonamy ich walidacji za pomocą badania kodu odpowiedzi. Jeśli kod odpowiedzi jest większy niż 400 uznajemy, że link jest niedziałający. Analogicznie jeśli link ma kod odpowiedzi mniejszy niż 400 uznajemy, że jest poprawny. Korzystamy z biblioteki HttpURLConnection.
Oczekiwany rezultat
Wyświetlamy listę linków wraz z informacją o stanie linka.
Warunki końcowe
Wyświetlana jest w konsoli lista 32 linków z informacją o ich stanie

TC7: Sprawdzenie poprawności działania pola wyszukiwania na stronie głównej

Warunki wstępne :
Wchodzimy na stronę główną projektu
http://sampleapp.tricentis.com/101/ .
Kroki testowe:
Klikamy na pole wyszukaj i wpisujemy “motor”. Zostajemy przekierowani na inną stronę. W tej chwili mamy dwa okna : okna główne(rodzica) i okno, które otworzyło się po wpisaniu słów w polu wyszukaj(okno typu child window). Za pomocą metody getWindowHandle sprawdzamy czy otworzyło się drugie okno. Przy pomocy iteratora i hasNext sprawdzamy kolejne okno i do niego przechodzimy.

Oczekiwany rezultat :
Sprawdzamy tytuł okna i adres url w asercji. Sprawdzane jest okno child window.
Warunki końcowe :
Tytuł okna i adres url odpowiadają stanu faktycznemu. Przekierowanie odbywa się na odpowiednią stronę.

3. Kod w Java

Kod został przesłany do folderu z projektami na grupie google oraz na adres email wykładowcy.
Projekt został wyeksportowany do pliku zip. Wraz z kodem projektu jest dołączony opis projektu w Word oraz w pliku Excel.

4. Wnioski (opcjonalne)

Projekt frameworka został napisany w edytorze Eclipse, z wykorzystaniem biblioteki Selenium w wersji 4.1.3, w Windows 10. Testy zostały napisane z wykorzystaniem biblioteki TestNG. W pliku config.properties są zawarte dane konfiguracyjne frameworka. Framework jest przystosowany na obsługę Chrome, Firefox, Internet Explorer. Projekt korzysta z DriverManagera. Domyślna przeglądarka to Chrome. Została we frameworku zastosowana biblioteka Extent reports , dzięki której możemy wygenerować raport w pliku html. W katalogu testData znajduje się plik z danymi z roszerzeniem xlsx. Do odczytu pliku w formacie pliku Excel wykorzystano bibliotekę Apache POI. Sterowanie testami jest za pomocą pliku testng.xml. Zależności, używane pluginy są w pliku pom.xml.

PROJEKT: AUTOMATYZACJA TESTÓW APLIKACJI WEBOWEJ W JAVA

UWAGI:

- Projekt można wykonać **dla dowolnej strony internetowej/aplikacji działającej w sieci**
- Na ocenę **4.0** trzeba zautomatyzować **5** przypadków testowych **bez zastosowania Page Object, Page Factory lub testowania opartego na danych**
- Na ocenę **5.0 należy dodatkowo zastosować wzorzec projektowy Page Object Pattern**
- Na ocenę **5.5 należy zastosować wzorzec Page Object i podejście DDT** (czytanie danych ze źródeł zewnętrznych tj. np. pliki .csv, baza danych itp., wykluczone są parametry ze suity testowej)
- Na zaliczenie trzeba wykonać **co najmniej jeden projekt z trzech.** Pozostałe projekty prowadzone są przez pozostałych prowadzących
- Czas wykonania: **do 1.06.2022**
- Pytania i uwagi odnośnie tego projektu proszę kierować po adres: **lzlocki@kozminski.edu.pl**