УДК 004.054

С. В. Кравченко[1]
І. В. Богач[1]

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ТЕСТУВАННЯ WEB-МАГАЗИНІВ

[1] Вінницький національний технічний університет

Анотація
Розглянуто найпопулярніші фреймворки для мови програмування Python, такі як PyTest, unittest, Robot, та виділено їх основні принципи роботи, переваги та недоліки.
Ключові слова: Python, веб-магазини, PyTest, unittest, Robot, тестування веб-магазинів.

Abstract
Considered the most popular frameworks for the Python programming language, such as PyTest, unittest, Robot, and their basic principles of operation, advantages and disadvatages.
Keywords: Python, web-stores, PyTest, unittest, Robot, web-store testing.

Вступ

На сьогоднішній день використання веб-магазинів є актуальною темою, оскільки багато людей щодня купують різні товари в інтернет-магазинах, при цьому не виходячи з дому. У період карантину, коли мільярди людей вимушені були сидіти вдома, зріс попит на онлайн-покупки і веб-магазини стали потрібні ще більше.

Інтернет зближує людей, реакція на будь-яку подію відбувається одразу, дистанція зникає. Веб-магазини значно знижують витрати для виробників, заощаджують на витратах на обслуговування звичайних магазинів, розширюють ринок та можливості покупців купувати будь-який товар у будь-якому місці у будь-який час доби. Це дає веб-магазинам перевагу перед звичайними магазинами. Також це важливо при переході виробників від «звичайної» до «електронної» торгівлі.

Розробка і тестування веб-магазинів тісно пов’язані між собою, тому що не можна випускати готовий продукт без попередніх тестів. Це може призвести до таких проблем як неправильне відображення сторінки чи товарів/послуг на ній, помилкове списання коштів користувача, неточний обрахунок суми товарів/послуг та інших. Тому тестування є важливою частиною процесу розробки.

Python є однією з найбільш використовуваних мов програмування у тестуванні. Однією з декількох причин популярності Python є широка підтримка фреймворків автоматизації тестування. Більшість популярних фреймворків Python сумісні з Selenium test automation framework та використовуються для автоматизації тестування браузерів та крос-браузерного тестування. У роботі розглянуто найпопулярніші фреймворки: PyTest, unittest та Robot [1].

Результати дослідження

PyTest [2] - це платформа тестування Python з відкритим кодом, яка в основному використовується для модульного тестування. Ця конкретна платформа тестування Python є масштабованою, оскільки вона корисна для написання простих тестів автоматизації, а також складних функціональних тестів для додатків і бібліотек. Набори тестів, написані за допомогою PyTest, є більш компактними, оскільки не потрібно багато стандартного коду і немає вимоги включати тести у великі тестові класи.

Переваги PyTest :

- тести які використовують PyTest нескладно зрозуміти, оскільки стандартного коду дуже мало;
- він сумісний з іншими платформами тестування Python, такими як unittest (або PyUnit);
- разом із простими тестами, PyTest також можна використовувати для створення складних функціональних тестових випадків.
- PyTest можна використовувати для проектів, які практикують TDD (Test Driven Development), а також проектів із відкритим кодом;

- для перенесення існуючих реалізацій за допомогою інших платформ тестування Python на PyTest потрібно менше зусиль;

- він підтримує прилади та класи, завдяки яким загальні тестові об'єкти доступні протягом життєвого циклу модуля/класу/функції/сеансу;

- PyTest підтримує паралельне виконання тестів через плагін pytest-xdist;

- Asserts в PyTest надають детальну інформацію про сценарії збою;

- він сприяє створенню ефективних тестових випадків (і наборів тестів), оскільки підтримує параметризацію. Використовуючи параметризацію, тестові випадки можуть виконуватися з різними конфігураціями введення, що призводить до мінімального повторення коду;

- PyTest є розширюваним, а платформа тестування Python вже має багату архітектуру плагінів. На даний момент існує понад 315 зовнішніх плагінів із PyTest.

Недоліки PyTest:

PyTest несумісний з іншими фреймворками тестування Selenium Python, оскільки використовує власні спеціальні підпрограми для розробки. Переписування повного коду - це єдиний можливий спосіб перенести існуючу реалізацію за допомогою PyTest на іншу систему тестування Python.

Незважаючи на недоліки, PyTest все ж таки має свою спільноту підтримки і використовується багатьма розробниками.

unittest (також званий PyUnit) [3] - це стандартна платформа тестування на Python, яка є частиною стандартної бібліотеки мови програмування. Python Testing Framework натхненний фреймворком JUnit. Це частина модуля unittest, яка постачається з моменту випуску версії Python 2.1.

Оскільки фреймворк тестування на Python для модульного тестування доступний одразу, багато розробників, які починають працювати з автоматизацією тестування Selenium, віддають перевагу фреймворку unittest.

Переваги unittest:

- додаткова установка пакета не потрібна;

- оскільки PyUnit є похідною від фреймворку xUnit і має явну схожість із цим фреймворком, новачкам розробникам у Python легко розпочати роботу з фреймворком unittest;

- звіт про тестування створюється дуже швидко і займає всього кілька мілісекунд;

- тестові сценарії можна виконувати незалежно або об'єднувати в TestSuite. Процес виконання простий, оскільки тести можна виконувати, лише згадуючи їх імена в терміналі;

- вивід виконання за замовчуванням є стислим і легким для розуміння. Ця платформа тестування Python також має ряд параметрів командного рядка, які допомагають надати більш детальний результат.

Недоліки unittest:

- назви тестових випадків все ще ґрунтуються на конвенції щодо найменування camelCase, що використовується в Java;

- для реалізації тесту використовується велика кількість шаблонного коду;

- немає можливості виведення кольорів.

Robot [4] - ще одна широко використовувана платформа тестування на Python, яка використовується для автоматизації тестування Selenium, RPA (Robotic Process Automation) і ATDD (Acceptance Test Driven Development). Це платформа для тестування на Python із відкритим вихідним кодом та можливістю розширювання. Фреймворк можна легко інтегрувати із будь-яким іншим інструментом, завдяки чому він ідеально підходить для створення гнучких рішень автоматизованого тестування.

Переваги Robot:

- вам не потрібні знання програмування, щоб писати тестові приклади за допомогою фреймворку Robot;

- розпочати роботу із фреймворком Robot легко, адже синтаксис тестових даних можна легко використовувати;

- оскільки фреймворк тестування на Python не залежить від платформи та програми, при перенесенні коду з однієї платформи на іншу не потрібно докладати додаткових зусиль;

- його можна використовувати для автоматизації тестування Selenium на основі BDD, ATDD та ключових слів;

- Robot має хороший вбудований механізм звітності, оскільки журнал HTML створюється після кожної збірки.

Недоліки Robot:

- немає вбудованої підтримки для паралельного виконання тестів;
- створення персоналізованих звітів HTML - складне завдання.

Висновки

Розглянуті в даній роботі фреймворки є найбільш популярними та призначені для розв'язання задач різніх типів. Вибір правильного фреймворку багато в чому залежить від вимог, які висуваються до розроблюваного програмного забезпечення. Тому вибір фреймворку напряму залежить від поставлених задач.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Top Python Testing Frameworks In 2020 For Selenium Automation [Електронний ресурс] – Режим доступу до ресурсу: https://www.codementor.io/@aadidwi/top-python-testing-frameworks-in-2020-for-selenium-automation-16fu6ll5ol.
2. pytest: документация на русском языке [Електронний ресурс] – Режим доступу до ресурсу: https://pytest-docs-ru.readthedocs.io/ru/latest/index.html.
3. unittest — Unit testing framework [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/library/unittest.html.
4. Robot Framework [Електронний ресурс] – Режим доступу до ресурсу: https://robotframework.org/.

Кравченко Софія Віталіївна *– студентка групи 2КН-18Б, кафедра комп'ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: emeli.braun1122000@gmail.com*

Богач Ілона Віталіївна *– к.т.н., доцент кафедри автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: ilona.bogach@gmail.com*

Kravchenko Sofia Vitaliivna *– student of 2CS-18B group, Department of Computer Science, Faculty of Intelligent Information Technology and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: emeli.braun1122000@gmail.com*

Bogach Ilona Vitaliivna *– Associate Professor of Automation and Intelligent Information Technologies, Faculty of Intelligent Information Technology and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: ilona.bogach@gmail.com*