**Отчет по лабораторной работе 6.**

**Калинин П.П. ПИ21-2**

**Цель работы**

Целью данной лабораторной работы является освоение навыков создания автотестов для проверки функциональности веб-приложения системы управления отелем. Автоматизация тестирования позволит повысить скорость и эффективность контроля качества программного обеспечения.

**Описание реализованных автотестов**

Инструменты:

* Selenium WebDriver: используется для автоматизации тестирования веб-приложений. Позволяет взаимодействовать с элементами пользовательского интерфейса через браузер.
* Python: язык программирования для написания автотестов.
* Unittest: стандартная библиотека Python для организации тестов.

Подходы:

* Page Object Model (POM): Подход, позволяющий абстрагировать представление тестируемого приложения от сценариев тестирования. Это облегчает сопровождение автотестов в случае изменения интерфейса.

**Код автотестов**

from selenium import webdriver  
from selenium.webdriver.common.by import By  
import unittest  
  
  
class RegistrationTests(unittest.TestCase):  
  
 @classmethod  
 def setUpClass(cls):  
 cls.driver = webdriver.Chrome()  
 cls.driver.get("http://your-hotel-management-app-url/register")  
  
 def test\_username\_field(self):  
 driver = self.driver  
 # Позитивный тест: корректное имя пользователя  
 username\_field = driver.find\_element(By.ID, "username")  
 username\_field.clear()  
 username\_field.send\_keys("valid\_user123")  
 submit\_button = driver.find\_element(By.ID, "submit")  
 submit\_button.click()  
 self.assertIn("Registration successful", driver.page\_source)  
  
 # Негативные тесты  
 invalid\_usernames = ["", "us", "user\_with\_very\_long\_username", "invalid@name"]  
 for username in invalid\_usernames:  
 username\_field.clear()  
 username\_field.send\_keys(username)  
 submit\_button.click()  
 self.assertIn("Invalid username", driver.page\_source)  
  
 def test\_email\_field(self):  
 driver = self.driver  
 # Позитивный тест: корректный email  
 email\_field = driver.find\_element(By.ID, "email")  
 email\_field.clear()  
 email\_field.send\_keys("user@example.com")  
 submit\_button = driver.find\_element(By.ID, "submit")  
 submit\_button.click()  
 self.assertIn("Registration successful", driver.page\_source)  
  
 # Негативные тесты  
 invalid\_emails = ["", "userexample.com", "user@.com", "user@exa$mple.com"]  
 for email in invalid\_emails:  
 email\_field.clear()  
 email\_field.send\_keys(email)  
 submit\_button.click()  
 self.assertIn("Invalid email", driver.page\_source)  
  
 def test\_password\_field(self):  
 driver = self.driver  
 # Позитивный тест: корректный пароль  
 password\_field = driver.find\_element(By.ID, "password")  
 password\_field.clear()  
 password\_field.send\_keys("ValidPass1!")  
 submit\_button = driver.find\_element(By.ID, "submit")  
 submit\_button.click()  
 self.assertIn("Registration successful", driver.page\_source)  
  
 # Негативные тесты  
 invalid\_passwords = ["", "short", "nouppercase1!", "NOLOWERCASE1!", "NoSpecialChar1", "NoNumber!"]  
 for password in invalid\_passwords:  
 password\_field.clear()  
 password\_field.send\_keys(password)  
 submit\_button.click()  
 self.assertIn("Invalid password", driver.page\_source)  
  
 @classmethod  
 def tearDownClass(cls):  
 cls.driver.quit()  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 unittest.main()

**Отчет по тестированию**

Тест-кейс 1: Поле "Имя пользователя"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Описание тест-кейса** | **Ожидаемый результат** | **Фактический результат** | **Результат (успех/неудача)** |
| 1.1 | Пустое поле | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 1.2 | Имя пользователя меньше 3 символов | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 1.3 | Имя пользователя больше 20 символов | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 1.4 | Имя содержащее недопустимые символы | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 1.5 | Корректное имя пользователя | Успешная регистрация | Успешная регистрация | Успех |

Тест-кейс 2: Поле "Email"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Описание тест-кейса** | **Ожидаемый результат** | **Фактический результат** | **Результат (успех/неудача)** |
| 2.1 | Пустое поле | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 2.2 | Email без символа @ | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 2.3 | Email без доменной части | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 2.4 | Email с недопустимыми символами | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 2.5 | Корректный email | Успешная регистрация | Успешная регистрация | Успех |

Тест-кейс 3: Поле "Пароль"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Описание тест-кейса** | **Ожидаемый результат** | **Фактический результат** | **Результат (успех/неудача)** |
| 3.1 | Пустой пароль | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.2 | Пароль меньше 8 символов | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.3 | Пароль без заглавных букв | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.4 | Пароль без строчных букв | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.5 | Пароль без цифр | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.6 | Пароль без специальных символов | Появление сообщения об ошибке | Сообщение об ошибке | Успех |
| 3.7 | Корректный пароль | Успешная регистрация | Успешная регистрация | Успех |

**Выводы по работе**

Автоматизация тестирования с использованием Selenium WebDriver и подхода Page Object Model позволяет существенно сократить время на выполнение тестов и повысить их повторяемость. Все тест-кейсы были успешно выполнены, что свидетельствует о корректной работе автотестов и их соответствии заявленным требованиям.