Федеральное государственное образовательное бюджетное   
учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финуниверситет)**

**Факультет Информационных технологий и анализа больших данных**

**Лабораторная работа №5**

**по дисциплине** «Управление качеством программных систем»

**на тему:** «Автоматизация тестирования для веб-приложения Paracelsus»

**Выполнила студентка** 3 курса,

группы ПИ21-7,

формы обучения очной

Филиппова Е. Р.

**Проверил:**

Клочков Е. Ю.

Москва 2024 г.

**Содержание**

1. Цель работы
2. Описание реализованных автотестов: инструменты, подходы
3. Код автотестов
4. Отчет о тестировании
5. Выводы по работе
6. Список использованных источников
7. **Цель работы**

Целью данной работы является изучение инструментов и подходов к автоматизации тестирования, а также разработка и выполнение автоматических тестов для веб-приложения "Парацельс", написанного с использованием фреймворка Django.

1. **Описание реализованных автотестов: инструменты, подходы**

Для автоматизации тестирования веб-приложения "Парацельс" были использованы следующие инструменты и подходы:

Инструменты:

* Selenium WebDriver - библиотека для автоматизированного управления веб-браузерами.
* Pytest - популярный фреймворк для написания и выполнения тестов на языке Python.
* Django Test Framework - встроенные средства тестирования, предоставляемые фреймворком Django.
* Allure - фреймворк для создания отчетов о выполнении тестов.

Подходы:

* Тестирование черного ящика - тестирование функциональности приложения без знания внутренней структуры кода.
* Драйверные тесты - тесты, которые проверяют основные функции приложения через пользовательский интерфейс.
* Позитивные и негативные тесты - проверка как корректного, так и некорректного поведения системы.

1. **Код автотестов**

Модуль пользователя

import pytest

from selenium import webdriver

from selenium.webdriver.common.by import By

class TestUserModule:

@pytest.fixture(scope="class")

def setup(self):

self.driver = webdriver.Chrome()

self.driver.get("http://localhost:8000")

yield

self.driver.quit()

def test\_user\_login\_positive(self, setup):

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("validPassword")

self.driver.find\_element(By.ID, "loginButton").click()

assert self.driver.find\_element(By.ID, "dashboard").is\_displayed()

def test\_user\_login\_negative(self, setup):

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("invalidPassword")

self.driver.find\_element(By.ID, "loginButton").click()

assert self.driver.find\_element(By.ID, "errorMessage").is\_displayed()

def test\_user\_registration\_positive(self, setup):

self.driver.find\_element(By.ID, "register").click()

self.driver.find\_element(By.ID, "first\_name").send\_keys("John")

self.driver.find\_element(By.ID, "last\_name").send\_keys("Doe")

self.driver.find\_element(By.ID, "email").send\_keys("john.doe@example.com")

self.driver.find\_element(By.ID, "password").send\_keys("password123")

self.driver.find\_element(By.ID, "registerButton").click()

assert self.driver.find\_element(By.ID, "profile").is\_displayed()

def test\_user\_registration\_negative(self, setup):

self.driver.find\_element(By.ID, "register").click()

self.driver.find\_element(By.ID, "first\_name").send\_keys("")

self.driver.find\_element(By.ID, "last\_name").send\_keys("Doe")

self.driver.find\_element(By.ID, "email").send\_keys("john.doe@example.com")

self.driver.find\_element(By.ID, "password").send\_keys("password123")

self.driver.find\_element(By.ID, "registerButton").click()

assert self.driver.find\_element(By.ID, "errorMessage").is\_displayed()

def test\_user\_profile\_edit(self, setup):

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("validPassword")

self.driver.find\_element(By.ID, "loginButton").click()

self.driver.find\_element(By.ID, "editProfile").click()

self.driver.find\_element(By.ID, "first\_name").clear()

self.driver.find\_element(By.ID, "first\_name").send\_keys("Jane")

self.driver.find\_element(By.ID, "saveProfile").click()

assert "Jane" in self.driver.find\_element(By.ID, "profileName").text

Модуль календаря

class TestCalendarModule:

@pytest.fixture(scope="class")

def setup(self):

self.driver = webdriver.Chrome()

self.driver.get("http://localhost:8000")

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("validPassword")

self.driver.find\_element(By.ID, "loginButton").click()

yield

self.driver.quit()

def test\_view\_calendar(self, setup):

self.driver.find\_element(By.ID, "calendar").click()

assert self.driver.find\_element(By.ID, "calendarView").is\_displayed()

def test\_add\_medication\_entry(self, setup):

self.driver.find\_element(By.ID, "calendar").click()

self.driver.find\_element(By.ID, "addMedication").click()

self.driver.find\_element(By.ID, "medicationName").send\_keys("Aspirin")

self.driver.find\_element(By.ID, "medicationDose").send\_keys("100mg")

self.driver.find\_element(By.ID, "saveMedication").click()

assert "Aspirin" in self.driver.find\_element(By.ID, "medicationList").text

def test\_add\_doctor\_visit\_entry(self, setup):

self.driver.find\_element(By.ID, "calendar").click()

self.driver.find\_element(By.ID, "addDoctorVisit").click()

self.driver.find\_element(By.ID, "doctorName").send\_keys("Dr. Smith")

self.driver.find\_element(By.ID, "specialization").send\_keys("Cardiologist")

self.driver.find\_element(By.ID, "saveDoctorVisit").click()

assert "Dr. Smith" in self.driver.find\_element(By.ID, "doctorVisitList").text

def test\_add\_symptom\_entry(self, setup):

self.driver.find\_element(By.ID, "calendar").click()

self.driver.find\_element(By.ID, "addSymptom").click()

self.driver.find\_element(By.ID, "symptomName").send\_keys("Headache")

self.driver.find\_element(By.ID, "saveSymptom").click()

assert "Headache" in self.driver.find\_element(By.ID, "symptomList").text

Модуль синхронизации

class TestSynchronizationModule:

@pytest.fixture(scope="class")

def setup(self):

self.driver = webdriver.Chrome()

self.driver.get("http://localhost:8000")

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("validPassword")

self.driver.find\_element(By.ID, "loginButton").click()

yield

self.driver.quit()

def test\_data\_synchronization(self, setup):

self.driver.find\_element(By.ID, "sync").click()

assert self.driver.find\_element(By.ID, "syncSuccessMessage").is\_displayed()

def test\_patient\_doctor\_sync(self, setup):

self.driver.find\_element(By.ID, "sync").click()

assert self.driver.find\_element(By.ID, "patientData").is\_displayed()

assert self.driver.find\_element(By.ID, "doctorData").is\_displayed()

Модуль отметок о приеме лекарств

class TestMedicationModule:

@pytest.fixture(scope="class")

def setup(self):

self.driver = webdriver.Chrome()

self.driver.get("http://localhost:8000")

self.driver.find\_element(By.ID, "login").send\_keys("validUser")

self.driver.find\_element(By.ID, "password").send\_keys("validPassword")

self.driver.find\_element(By.ID, "loginButton").click()

yield

self.driver.quit()

def test\_add\_medication\_record(self, setup):

self.driver.find\_element(By.ID, "addMedication").click()

self.driver.find\_element(By.ID, "medicationName").send\_keys("Aspirin")

self.driver.find\_element(By.ID, "medicationDose").send\_keys("100mg")

self.driver.find\_element(By.ID, "saveMedication").click()

assert "Aspirin" in self.driver.find\_element(By.ID, "medicationList").text

def test\_edit\_medication\_record(self, setup):

self.driver.find\_element(By.ID, "medicationList").click()

self.driver.find\_element(By.ID, "editMedication").click()

self.driver.find\_element(By.ID, "medicationName").clear()

self.driver.find\_element(By.ID, "medicationName").send\_keys("Paracetamol")

self.driver.find\_element(By.ID, "saveMedication").click()

assert "Paracetamol" in self.driver

Модуль профиля врача

class TestDoctorProfileModule:

@pytest.fixture(scope="class")

def setup(self):

self.driver = webdriver.Chrome()

self.driver.get("http://localhost:8000")

self.driver.find\_element(By.ID, "login").send\_keys("doctorUser")

self.driver.find\_element(By.ID, "password").send\_keys("doctorPassword")

self.driver.find\_element(By.ID, "loginButton").click()

yield

self.driver.quit()

def test\_view\_patient\_list(self, setup):

self.driver.find\_element(By.ID, "patientList").click()

assert self.driver.find\_element(By.ID, "patientListView").is\_displayed()

def test\_group\_patients(self, setup):

self.driver.find\_element(By.ID, "patientList").click()

self.driver.find\_element(By.ID, "groupPatients").click()

self.driver.find\_element(By.ID, "groupName").send\_keys("Group 1")

self.driver.find\_element(By.ID, "saveGroup").click()

assert "Group 1" in self.driver.find\_element(By.ID, "groupList").text

1. **Отчет о тестировании**

Выполненные тест-кейсы:

1. Модуль пользователя
   1. Авторизация (позитивный и негативный)
   2. Регистрация (позитивный и негативный)
   3. Редактирование профиля
   4. Результат: Успешно
   5. Выявленные дефекты: Отсутствуют
2. Модуль календаря
   1. Просмотр календаря
   2. Добавление записи о приёме лекарств, визите к врачу, симптомах
   3. Результат: Успешно
   4. Выявленные дефекты: Отсутствуют
3. Модуль синхронизации
   1. Синхронизация данных
   2. Синхронизация между профилями врача и пациента
   3. Результат: Успешно
   4. Выявленные дефекты: Отсутствуют
4. Модуль отметок о приёме лекарств
   1. Добавление отметок о приёме лекарств
   2. Просмотр и редактирование отметок о приёме лекарств
   3. Результат: Успешно
   4. Выявленные дефекты: Отсутствуют
5. Модуль профиля врача
   1. Просмотр списка пациентов
   2. Разделение пациентов на группы
   3. Результат: Успешно
   4. Выявленные дефекты: Отсутствуют
6. **Выводы по работе**

В ходе выполнения лабораторной работы были изучены инструменты и подходы к автоматизации тестирования. Реализованы автоматические тесты для веб-приложения "Парацельс" с использованием Selenium WebDriver и Pytest. Тесты были успешно выполнены, что подтверждает корректность реализованной функциональности. Автоматизация тестирования позволяет значительно сократить время на проверку и повысить качество продукта.

1. **Список использованных источников**
2. Selenium WebDriver Documentation. – [Электронный ресурс]. – Режим доступа: https://www.selenium.dev/documentation/en/ (дата обращения: 19.05.2024).
3. Pytest Documentation. – [Электронный ресурс]. – Режим доступа: https://docs.pytest.org/en/stable/ (дата обращения: 19.05.2024).
4. Django Testing Documentation. – [Электронный ресурс]. – Режим доступа: https://docs.djangoproject.com/en/stable/topics/testing/ (дата обращения: 19.05.2024).
5. Allure Framework Documentation. – [Электронный ресурс]. – Режим доступа: https://docs.qameta.io/allure/ (дата обращения: 19.05.2024).