МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по производственной практике

Тема: Selenium тесты для
кафедрального сервиса опросов

Студент гр. 8303 _____ Спирин Н.В.

Руководитель _____ Берленко Т.А.

Санкт-Петербург
2021

ЗАДАНИЕ

НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ

Студент Спирин Н.В.

Группа 8303

Тема практики: Selenium тесты для кафедрального сервиса опросов

Задание на практику:

Написать тесты для опроса с использованием Selenium и Docker, а также с применением паттерна Page Object.

Сроки прохождения практики: 04.03.2021 – 12.07.2021

Дата сдачи отчета: 09.07.2021

Дата защиты отчета: 09.07.2021

Студент _____ Спирин Н.В.

Руководитель _____ Берленко Т.А.

АННОТАЦИЯ

Целью прохождения практики является знакомство с технологиями Selenium и Docker и обучение навыку написания автоматических тестов.

SUMMARY

The purpose of the internship is to get acquainted with Selenium and Docker technologies and learn the skill of writing automatic tests.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Главной задачей практики является написание тестов, успешное прохождение которых будет гарантировать работоспособность тестируемого кафедрального сервиса опросов. К побочным задачам можно отнести нахождение уже существующих ошибок в работе опроса и информирование о них руководителя практики для их последующего исправления.

1. РАЗДЕЛЕНИЕ ЛОГИКИ И РЕАЛИЗАЦИИ. PAGE OBJECT

Для написания selenium тестов для кафедрального сервиса опросов был использован шаблон проектирования Page Object, который позволяет разделять логику выполнения тестов от их реализации. Page Object как бы моделирует страницы тестируемого приложения в качестве объектов в коде. В результате его использования получаются отдельные классы, отвечающие за работу с HTML каждой конкретной веб-страницы. Такой подход значительно уменьшает объем повторяющегося кода, потому что одни и те же объекты страниц можно использовать в различных тестах.

Существует большая разница между логикой тестирования (что проверить) и его реализацией (как проверить). Пример тестового сценария: «Пользователь вводит неверный логин или пароль, нажимает кнопку входа, получает сообщение об ошибке». Этот сценарий описывает логику теста, в то время как реализация содержит в себе такие действия как поиск полей ввода на странице, их заполнение, проверку полученной ошибки и т.д. И если, например, изменится способ вывода сообщения об ошибке, то это никак не повлияет на сценарий теста, все также нужно будет ввести неверные данные, нажать кнопку входа и проверить ошибку. Но это напрямую затронет реализацию теста — необходимо будет изменить метод получающий и обрабатывающий сообщение об ошибке. При разделении логики теста от его реализации автотесты становятся более гибкими и их, как правило, легче поддерживать.

2. РЕАЛИЗАЦИЯ ТЕСТИРОВАНИЯ

Для реализации тестирования были созданы классы start_page, survey_page и finish_page. Данные классы отвечают за удобную работу с соответствующими страницами опроса.

2.1. Методы класса start_page

def count_directions(self) – возвращает количество направлений, которые можно выбрать

def choose_direction(self, num_direction) – выбирает направление, переданное функции в качестве аргумента - num_direction

def choose_sem(self, num_sem) – выбирает семестр, переданный функции в качестве аргумента - num_sem

def write_id(self, input_id) – вводит в поле id значение input_id

def click_start_button(self) – производит клик по кнопке старта опроса

def click_start_button_success(self) – обрабатывает успешное нажатие на кнопку старта опроса

def click_start_button_alert(self) – обрабатывает нажатие на кнопку старта опроса в случае допущения пользователем ошибки при вводе

def find_alert(self) – находит предупреждение, которое появилось после ошибки ввода

2.2. Методы класса survey_page

def is_compulsory_rating_current_subject(self, rating) – определяет, является ли рейтинг текущего предмета обязательным для заполнения

def get_teacher_names(self) – возвращает список имен всех преподавателей на текущей странице

def go_to_next_teacher(self) – находит и делает текущим следующий предмет в опросе

def go_to_next_teacher_in_menu(self) - находит и делает текущим следующий предмет в меню опроса

def choose_rating_relevance(self, rating) – ставит оценку актуальности, равную rating, текущему предмету

def choose_rating_quality(self, rating) – ставит оценку качеству, равную rating, текущему предмету

def click_finish_button(self) – осуществляет клик по кнопке завершения опроса

def click_finish_button_success(self) – обрабатывает успешное нажатие на кнопку завершения опроса

def click_finish_button_alert(self) – обрабатывает нажатие на кнопку завершения опроса в случае допущения пользователем ошибки при вводе

def find_alert(self) – находит предупреждение, которое появилось после ошибки валидации

2.3. Методы класса finish_page

def get_final_text(self) – находит текст на последней странице, который определяет прохождение пользователем опроса

3. ЛОГИКА ТЕСТИРОВАНИЯ

Для реализации логики были созданы классы test_start_page и test_survey_page. Данные классы занимаются непосредственно тестированием опроса.

3.1. Методы класса test_start_page

def tearDown(self) – запускается автоматически после каждого теста, инициализируя стартовую страницу заново

def test_choosing_directions(self) – проверяет успешность выбора всех направлений

def test_choosing_semesters_from_first_direction(self) – проверяет успешность выбора семестров после выбора первого направления

def test_choosing_semesters_from_second_direction(self) – проверяет успешность выбора семестров после выбора второго направления

def test_choosing_semesters_from_third_direction(self) – проверяет успешность выбора семестров после выбора третьего направления

def test_choosing_semesters_from_fourth_direction(self) – проверяет успешность выбора семестров после выбора четвертого направления

def test_input_id(self) – проверяет успешность ввода идентификатора

def test_alert_choosing_sem(self) – проверяет наличие правильного предупреждения в случае если пользователь забыл выбрать семестр

def test_alert_filling_id(self) – проверяет наличие правильного предупреждения в случае если пользователь забыл ввести идентификатор

def test_start_survey(self) – проверяет успешность перехода на страницу опроса в случае если пользователь не ошибся при вводе

3.2. Методы класса test_survey_page

def tearDown(self) – запускается автоматически после каждого теста, инициализируя стартовую страницу заново

def choose_direction_and_semester(self, direction, semester) – выбирает направление и семестр, переданные функции в качестве аргументов, вводит тестовый id и проверяет успешность нажатия на кнопку старта опроса

def validation_survey_alert(self, direction, semester) – сначала вызывает функцию choose_direction_and_semester(self, direction, semester) для перехода на страницу опроса соответствующего номера направления и семестра, а затем сразу нажимает на кнопку завершения опроса. После этого проверяется наличие предупреждения о том, что обязательные поля не были заполнены

def validation_survey_success(self, direction, semester) – сначала вызывает функцию choose_direction_and_semester(self, direction, semester) для перехода на страницу опроса соответствующего номера направления и семестра. Затем функция проходит по всей странице в поиске повторяющихся предметов и преподавателей и выдает ошибку в случае их обнаружения. После этого заполняются все обязательные поля и проверяется успешность прохождения опроса

def validation_survey_menu(self, direction, semester) – сначала вызывает функцию choose_direction_and_semester(self, direction, semester) для перехода на страницу опроса соответствующего номера направления и семестра. Затем проверяется уникальность имен меню на странице опроса.

def test_validation_survey_success_all(self) – вызывает функцию validation_survey_success(direction, sem) для всех направлений и семестров

def test_validation_survey_alert_all(self) – вызывает функцию validation_survey_alert(direction, sem) для всех направлений и семестров

def test_validation_survey_menu_all(self) – вызывает функцию validation_survey_menu(direction, sem) для всех направлений и семестров

ЗАКЛЮЧЕНИЕ

В ходе прохождения производственной практики были получены навыки по работе с Selenium и Docker. В итоге были написаны тесты для кафедрального сервиса опросов, а также обнаружены некоторые ошибки в работе опроса, о которых было сообщено руководителю практики.