OTUS

Специальная цена

Python QA Engineer

Курс по активной прокачке навыков программирования на Python для QA-инженеров

Длительность курса: 138 академических часов

1 Введение в автоматизацию тестирования

1	**Введение в разработку и тестирование**	создать свой проект на github, в который будут выкладываться ДЗ; проанализировать виды, цели, инструменты и инфраструктуру для автоматизированного тестирования.
2	**Введение в Pytest**	запускать тесты; писать параметризованные тесты; пользоваться фикстурами; добавить в проект библиотеку pytest.
3	**Работа с тестовыми данными**	работать с файлами основных типов (csv, xml, json): познакомиться с менеджерами контекста. Домашние задания 1 PyTest и работа с тестовыми данными Цель: Узнать особенности PyTest, научиться работать с различными типами файлов, написать тесты для pytest и собственный контекстный менеджер для файлов. Задание 1. Пишем первый пул тестов Написать по 5 авто-тестов на на методы каждой из структур данных в Python: 1) List 2) Set 3) Dictionary 4) String

Как минимум один тест для каждой структуры должен быть параметризован.
Для тестов на каждую структуру данных создать отдельный файл.

За основу взять методы которые есть у структур данных: https://docs.python.org/3.6/tutorial/datastructures.html#tuples-and-sequences

Задание 2. Работа с тестовыми данными

1. Скачать приложенные к занятию файлы форматов users.json и books.csv
2. Написать с помощью контекстных менеджеров скрипт, которых из двух файлов будет читать данные и на их основании создаст json файл со структурой из файла example.json
3. Каждому пользователю нужно добавить одну книгу из списка. Если количество книг меньше количества пользователей, то остальным добавить пустой массив. Если книг больше чем пользователей, то просто прекратить раздавать книги.

---

4 **Тестирование API. Rest, restful, SOAP, типы запросов**

тестировать REST API-сервисы.

Домашние задания

1 Тестирование API

Цель: Тестирование API сервиса с помощью Python используя библиотеки pytest, requests, json.

Тестирование каждого api оформить в отдельном тестовом модуле.

1. Тестирование REST сервиса 1
Написать минимум 5 тестов для REST API сервиса: https://dog.ceo/dog-api/.
Как минимум 2 из 5 должны использовать параметризацию.
Документация к API есть на сайте.
Тесты должны успешно проходить.

2. Тестирование REST сервиса 2
Написать минимум 5 тестов для REST API сервиса: https://www.openbrewerydb.org/.
Как минимум 2 из 5 должны использовать параметризацию.
Документация к API есть на сайте.
Тесты должны успешно проходить.

3. Тестирование REST сервиса 3.
Написать минимум 5 тестов для REST API сервиса: https://jsonplaceholder.typicode.com/.
Как минимум 2 из 5 должны использовать параметризацию.
Документация к API есть на сайте.
Тесты должны успешно проходить.

4. Реализуйте в отдельном модуле (файле) тестовую функцию которая будет принимать 2 параметра:
url - должно быть значение по умолчанию https://ya.ru
status_code - значение по умолчанию 200
Параметры должны быть реализованы через pytest.addoption.
Можно положить фикстуру и тестовую функцию в один файл.
Основная задача чтобы ваш тест проверял по переданному урлу статус ответа тот который передали,
т.е. по адресу https://ya.ru/sfhfhfhfhfhfhfhfh должен быть валидным ответ 404

пример запуска pytest test_module.py --url=https://mail.ru --status_code=200

---

5 **DDT в тестировании API**

тестировать с большим количеством данных;
применять DDT подход в тестировании API.

---

6 **Погружение в Python. ООП**

разобрать реализацию основных понятий ООП в Python.

Домашние задания

1 Реализуем ООП на практике

Цель: Научиться работать с парадигмой ООП, потренировать объектное мышение, поучиться писать тесты.

Создать класс геометрической фигуры.

Реализовать на его основе классы фигур Треугольник, Прямоугольник, Квадрат, Круг.

1 Часть.

Фигура должна иметь атрибуты:
name - название фигуры,
area - выводить площадь,
angles - выводить количество углов
perimeter - выводить периметр (сумму длин сторон, длину окружности)

Фигура должна реализовать метод add_square() который должен принимать другую фигуру и выводить сумму площадей этих фигур. Если передана не геометрическая фигура, то нужно выдавать ошибку и сообщать что передан неправильный класс.

2. Часть.

Написать тесты с использованием pytest на эти классы. По одному тесту на каждый метод каждой фигуры. Т.е. будет четыре тестовых модуля по 5 тестов на каждый. Можно написать и больше. :)

Задача: Потренировать объектно-ориентированное

		мышление, и написание тестов на собственный код.
7	**Погружение в Python: дескрипторы и статическая типизация, функциональное программирование**	сформировать представление о внутреннем устройстве языка; проанализировать концепцию функционального программирования.

2 Тестирование UI

1	**Введение в тестирование Web UI, Selenium WebDriver**	запускать и останавливать браузеры с помощью Selenium; готовить инфрастуктуру для запуска ui тестов. Домашние задания 1 Основы Selenium Цель: Научиться настраивать окружение для Selenium тестов, написать тесты, настроить ожидания к проекту. Научиться писать простые selenium скрипты. 1. Настроить Selenium для запуска тестов 2. Написать фикстуру для запуска трех разных браузеров (ie, firefox, chrome) в полноэкранном режиме с опцией headless. Выбор браузера должен осуществляться путем передачи аргумента командной строки pytest. По завершению работы тестов должно осуществляться закрытие браузера. 3. Добавить опцию командной строки, которая указывает базовый URL opencart. 4. Написать тест, который открывает основную страницу opencart (http://<ip_or_fqdn>/opencart/) и проверяет, что мы находимся именно на странице приложения opencart. 5. Написать тесты проверяющие наличие элементов на разных страницах приложения opencart. Реализовать минимум пять тестов (одни тест = одна страница приложения) Какие элементы проверять определить самостоятельно, но не меньше 5 для каждой страницы. Покрыть нужно: Главную / Каталог /index.php?route=product/category&path=20 Карточку товара /index.php?route=product/product&path=57&product_id=49 Страницу логина /index.php?route=account/login Страницу логина в админку /admin/ 6. К существующим тестам добавить явные ожидания элементов. 7. Реализовать 2 тестовых сценария на раздел администратора 7.1 Добавить проверку логина и разлогина раздела. 7.2 Добавить проверку перехода к разделу с товарами, что появляется таблица с товарами.
2	**Поиск и действия с элементами**	искать элементы с помощью Selenium и проводить с ними простые действия.
3	**Ожидания элементов**	работать с ожиданиями элементов.

4	**JavaScript in Selenium**	сделать сложные действия в selenium; проанализировать свойства объекта WebElement.
5	**Архитектура веб-тестов (Page Object, Page Element)**	реализовывать paje object модель. Домашние задания 1 PageObject Пишем тесты в паттерне PageObject.
6	**Работа с окнами**	работать с несколькими окнами одновременно.
7	**Протоколирование и отчетность**	логировать действия Selenium; переопределять стандартный листенер. Домашние задания 1 Инфраструктура для автотестов Цель: Научиться настраивать логирование, запуск на selenoid и составление отчетов в allure 1. Настройте логирование 2. Подключите allure и добавьте аннотации allure в тестах 3. Настройте Selenoid, добавьте несколько браузеров и запустите на нем тесты
8	**Удаленный запуск (Grid)**	запускать тесты на удаленной машине; запустить тесты в облаке.
9	**Selenoid**	объяснить как Selenoid решает проблему запуска тестов.
10	**Allure**	строить отчёты по результатам тестирования.

3 Работа с бекендом

1	**Анализ логов веб-сервера**	выявлять ошибки в бекенде; использовать утилиты grep, awk, find.
2	**Работа с БД**	работать с БД, использовать БД для работы с тестовыми данными и логами. Домашние задания 1 Анализ логов веб-сервера Цель: Научитесь анализировать логи веб-сервера В качестве источника логов взять логи opencart. 1. Написать скрипт анализа access.log для apache или nginx Требования к реализации 1. Должна быть возможность указать директорию где искать логи или конкретный файл 2. Должна быть возможность выбрать все файлы логов отпарсить или только заданный 3. В случае если файл не может быть обработан, то скрипт должен завершится с ошибкой 4. Для access.log должна собираться следующая информация: - общее количество выполненных запросов - количество запросов по типу: GET - 20, POST - 10 и т.п. - топ 10 IP адресов, с которых были сделаны запросы - топ 10 самых долгих запросов, должно быть видно метод, url, ip, время запроса - топ 10 запросов, которые завершились клиентской ошибкой, должно быть видно метод, url, статус код, ip адрес - топ 10 запросов, которые завершились ошибкой со стороны сервера, должно быть видно метод, url, статус код, ip адрес 5. Собранная статистика должна быть сохранена в json файл 6. Должен быть README файл, который описывает как работает скрипт
3	**Работа с сетью I**	работать с сетевыми протоколами прикладного уровня. Домашние задания 1 Работа с сетью Задани 1. Работа с сетью. Протоколы прикладкого уровня. Тесты бэкенд с использованием SSH клиента 1) Пишем код, который будет осуществлять подключения по SSH и работать с FTP. 2) Добавить тесты, которые используют SSH клиент для реализации следующих сценариев: перезагрузка сервера opencart с последующей проверкой, что opencart доступен, рестарт основных сервисов для opencart с последующей

		проверкой, что сервис доступен. Задание 2. Работа с сетью. Протоколы низкого уровня. Парсинг html страницы средствами python. 1) Пишем собственный HTTP клиент с использованием библиотеки socket. 2) Необходимо расширить предыдущее домашнее задание по парсингу заголовков HTTP функциональностью парсинга тела ответа. Задание 3. Работа с ОС Linux с помощью Python. Тесты, которые работают с сетевой, файловой и системой управления процессами Linux.
4	**Работа с сетью II**	работать с сетевыми протоколами низкого уровня; написать http-запросы; углубить знания в области работы сетей и веб-приложений.
5	**Архитектура Линукс**	работать с инструментами диагностики неисправностей Linux; работать с утилитами ping, netstat, ip.; диагностировать проблемы на уровне сети.
6	**Работа с ОС Linux с помощью Python**	работать с операционной системой Linux средствами Python.
7	**Траблшутинг**	находить причины неисправностей, которые возникли в ходе работы Linux.

4 DevOps

1	**Виртуализация. Контейнеры**	собирать собственные Docker контейнеры. Домашние задания 1 Подготовка тестовых данных для автотестов путём создания сущности в БД Написать коннектор к базе данных OpenCart Создать любую сущность через БД и написать тест проверки её создания (через селениум)
2	**Виртуализация. Виртуальные машины**	разворачивать и запускать виртуальные машины с помощью python.
3	**Непрерывная интеграция, Jenkins**	установить и настроить Jenkins CI.
4	**Подготовка тестового окружения**	подготовить тестовое окружение, соберать deb и whl пакеты.

5 Другие виды тестирования

1 **BDT часть 1**

написать тесты на RobotFramework.

Домашние задания

1 BDD

Цель: Задание 1. BDD. Использование пользовательских сценариев и RobotFramework
1) Научиться использовать подходы BDD для написания тест-кейсов.
2) Научиться запускать тесты RobotFramework

Задание 2. Расширенное использование RobotFramework и расширение библиотек RobotFramework на Python
1) Научиться использовать дополнительные библиотеки для RobotFramework
2) Научиться писать собственные модули для RobotFramework на Python

Задание 1. BDD. Использование пользовательских сценариев и RobotFramework
1) Написать сценарии использования в формате BDD(gherkin) для пользовательской части opencart (купить товар и т.д.)
2.1 Настроить работу с RobotFramework
2. 2 Написать тесты для администраторской панели на RobotFramework (выбрать самостоятельно 5 кейсов)

Задание 2. Расширенное использование RobotFramework и расширение библиотек RobotFramework на Python
1.
0 (*). Подключить библиотеку сохранения результатов тестов DbBot (https://github.com/robotframework/DbBot)
1.1. Подключить к тестам библиотеку DatabaseLibrary
1.2. Написать тест-кейс в RobotFramework создания нового товара в администраторской панели OpenCart
1.3. Сделать в этом же тесте проверку что новый товар существует
1.4. Во втором тест-кейсе удалить товар
1.5. Проверить методом count что количество товаров стало меньше
1.6. Должны быть сделаны скриншоты каждого шага в браузере

2. Создать тестсьюит на RobotFramework по тестированию логина (в админку и в клиентское приложение)
2.1. Создать файл .py с реализацией класса ЛогинАдмин (метод - логин в админ панель)
2.2. Создать файл .py с реализацией класса ЛогинКлиент (метод - логин в клиентскую часть)
2.3. Создать файл mylib.py, агрегирующий (1) и (2) (как в лекции)
2.4. Создать тестсьюит логина использующий библиотеку (3)
2.5. Написать два теста, один - для теста логина в админку, второй - для теста логина клиента

2	**BDT часть 2**	написать собственные расширения для RobotFramework.
3	**Основы безопасности веб-приложений**	проводить инструментальный анализ защищённости; писать тесты на проверку защищённости приложения.
4	**Mock**	пользоваться библиотекой Mock для написания собственных заглушек. Домашние задания 1 Модульное тестирование. Mock Objects. Цель: Научиться писать модульные тесты и использовать Mock объекты. Пишем модульные тесты в стиле xunit используя MockObjects. Для домашнего задания с тестированием REST API необходимо замокать запросы к REST сервису и его ответы.
5	**Модульное тестирование**	написать модульные тесты, используя библиотеку unittest.
6	**Нагрузочное тестирование**	написать нагрузочные тесты для веб-приложения.

6 Проектный модуль

1	**Выбор темы и организация проектной работы**	выбрать и обсудить тему проектной работы; спланировать работу над проектом; ознакомиться с регламентом работы над проектом.
2	**Собеседование Python QA, разбор тестовых заданий**	разобрать требования к qa automation на рынке труда в России; обсудить основные вопросы, которые задают на собеседовании.
3	**Консультация по проектам и домашним заданиям**	получить ответы на вопросы по проекту, ДЗ и по курсу. Домашние задания 1Проект
4	**Защита проектных работ**	защитить проект и получить рекомендации экспертов.
5	**Подведение итогов курса**	узнать, как получить сертификат об окончании курса, как взаимодействовать после окончания курса с OTUS и преподавателями, какие вакансии и позиции есть для выпускников (опционально - в России и за рубежом) и на какие компании стоит обратить внимание.